aws

使用者指南

Application Auto Scaling

Application Auto Scaling: 使用者指南

Table of Contents

什麼是 Application Auto Scaling？

Application Auto Scaling 是一項適用於開發人員和系統管理員的 Web 服務，這些開發人員和系統管理員需要解決方案來自動擴展個別 AWS 服務的可擴展性資源，而非 Amazon EC2 Auto Scaling。使用 Application Auto Scaling，您可以為下列資源設定自動擴展：

- AppStream 2.0 機群
- Aurora 複本
- Amazon Comprehend 文件分類和實體識別器端點
- DynamoDB 資料表和全域次要索引
- Amazon ECS 服務
- ElastiCache 複寫群組 (Redis OSS 和 Valkey) 和 Memcached 叢集
- Amazon EMR 叢集
- Amazon Keyspaces (適用於 Apache Cassandra) 資料表
- Lambda 函數佈建並行
- Amazon Managed Streaming for Apache Kafka (MSK) 代理程式儲存
- Amazon Neptune 叢集
- SageMaker AI 端點變體
- SageMaker AI 推論元件
- SageMaker AI Serverless 佈建並行
- Spot 機群請求
- Amazon WorkSpaces 集區
- 由您自家的應用程式或服務所提供的自訂資源。更多詳細資訊，請參閱 GitHub 儲存庫。

若要查看上述任何 AWS 服務的區域可用性，請參閱區域 tableRegion。

如需有關使用 Auto Scaling 群組來擴展 Amazon EC2 執行個體機群的詳細資訊，請參閱《Amazon EC2 Auto Scaling 使用者指南》。

Application Auto Scaling 的功能

Application Auto Scaling 可以根據您定義的條件，自動擴展可擴展的資源。

- 目標追蹤擴展 - 根據特定 CloudWatch 指標的目標值擴展資源。
- 步驟擴展 - 根據一組依警示違規程度而變動的擴展調整值擴展資源。
- 排定擴展 - 僅擴展一次或按照排定重複擴展資源。
- 預測擴展 – 根據歷史資料主動擴展資源，以符合預期的負載。

可搭配 Application Auto Scaling 使用

您可以使用以下介面設定擴展，使用哪個介面取決於要擴展的資源：

- AWS Management Console - 提供 Web 界面，讓您用來設定擴展。註冊 AWS 帳戶並登入 AWS Management Console。然後，針對簡介中列出的其中一個資源開啟服務主控台。例如，若要擴展 Lambda 函數，請開啟 AWS Lambda console。請確定您在與您要使用的資源相同的 AWS 區域 中開啟主控台。

  > **Note**
  >
  > 並非所有資源皆可透過主控台存取。如需詳細資訊，請參閱AWS 服務 可與 Application Auto Scaling 搭配使用。

- AWS Command Line Interface (AWS CLI) – 為廣泛的 提供命令 AWS 服務，並在 Windows、macOS 和 Linux 上支援 。若要開始使用，請參閱AWS Command Line Interface。如需命令清單，請參閱《 AWS CLI 命令參考》中的 application-autoscaling。
- AWS Tools for Windows PowerShell – 我們也為在 PowerShell 環境中編寫指令碼的使用者提供許多 AWS 產品的命令。若要開始使用，請參閱《AWS Tools for PowerShell 使用者指南》。如需詳細資訊，請參閱《AWS Tools for PowerShell Cmdlet 參考》。
- AWS SDKs – 提供特定語言的 API 操作，並負責許多連線詳細資訊，例如計算簽章、處理請求重試和處理錯誤。如需詳細資訊，請參閱要建置的工具 AWS。
- HTTPS API – 提供您可以使用 HTTPS 請求呼叫的低層級 API 動作。如需詳細資訊，請參閱《Application Auto Scaling API 參考》。
- AWS CloudFormation - 支援使用 CloudFormation 範本設定擴展。如需詳細資訊，請參閱使用 設定 Application Auto Scaling 資源 AWS CloudFormation。

若要以程式設計方式連線至 AWS 服務，您可以使用 端點。如需 Application Auto Scaling 呼叫端點的相關資訊，請參閱《 機密區域使用者指南》中的中國 Amazon Web Services 的端點和 ARN》中的 Application Auto Scaling 端點和配額AWS 一般參考《 機密區域使用者指南

Application Auto Scaling 概念

本主題說明主要概念，協助您瞭解並開始使用 Application Auto Scaling。

可擴展的目標

您建立的實體，用來指定您要擴展的資源。每個可擴展的目標都由服務命名空間、資源 ID 和可擴展的維度來唯一識別，代表基礎服務的某些容量維度。例如，Amazon ECS 服務支援自動擴展任務計數，DynamoDB 資料表支援自動擴展資料表及其全域次要索引的讀和寫容量，Aurora 叢集支援擴展複本計數。

> **Tip**
>
> 每個可擴展的目標也有容量下限和上限。擴展政策永遠不會高於或低於上下限範圍。您可以直接對基礎資源進行超出此範圍的變更，而 Application Auto Scaling 並不知情。不過，只要叫用擴展政策或呼叫 `RegisterScalableTarget` API，Application Auto Scaling 就會擷取目前的容量，並與容量下限和上限相比較。如果落在上下限範圍之外，則會將容量更新為符合設定的上限和下限。

縮減

當 Application Auto Scaling 自動減少可擴展目標的容量時，就稱為可擴展的目標「縮減」。設定擴展政策時，它們無法在低於其最小容量的可擴展目標中進行縮減。

擴展

當 Application Auto Scaling 自動增加可擴展目標的容量時，就稱為可擴展的目標「水平擴展」。設定擴展政策時，它們無法橫向擴展高於其最大容量的可擴展目標。

擴展政策

擴展政策會指示 Application Auto Scaling 追蹤特定的 CloudWatch 指標。然後，當指標高於或低於特定閾值時，決定採取什麼擴展動作。例如，您可能想在叢集的 CPU 使用率開始上升時水平擴展，而於再次下降時縮減。

用於自動擴展的指標由目標服務發佈，但您也可以將自己的指標發佈至 CloudWatch，然後用於擴展政策。

擴展活動之間的冷卻時間可在另一個擴展活動開始之前，先讓資源穩定。在冷卻時間，Application Auto Scaling 會持續評估指標。冷卻時間結束時，擴展政策會視需要啟動另一個擴展活動。在冷卻時間，根據目前的指標值，如果需要更大的水平擴展，擴展政策會立即水平擴展。

排定的動作

排定的動作會在特定日期和時間自動擴展資源。做法是修改可擴展目標的容量上限和下限，因此可用來調高容量下限或調低容量上限，以依據排程而縮減和水平擴展。例如，若應用程式在週末不耗用資源，您可以使用排定的動作在週五減少容量，然後在下週一增加容量，以此來擴展應用程式。

您也可以使用排定的動作來隨著時間最佳化最小值和最大值，以順應預期有高於正常流量的情況，例如行銷活動或季節性波動。這樣可協助您因為使用量增加而需要提高水平擴展時改善效能，並在使用較少的資源時降低成本。

進一步了解

AWS 服務 可與 Application Auto Scaling 搭配使用 - 本節介紹您可以擴展的服務，並協助您註冊可擴展的目標來設定自動擴展。也說明 Application Auto Scaling 為了存取目標服務中的資源，而建立的每個 IAM 服務連結角色。

Application Auto Scaling 的目標追蹤擴展政策 - Application Auto Scaling 的主要功能之一是目標追蹤擴展政策。瞭解目標追蹤政策如何根據您設定的指標和目標值，自動調整所需的容量，將使用率保持在一定水平。例如，您可以設定目標追蹤，將 Spot 機群的 CPU 平均使用率維持在 50%。然後，Application Auto Scaling 會視需要啟動或終止 EC2 執行個體，將所有伺服器的整體 CPU 使用率維持在 50%。

AWS 服務 可與 Application Auto Scaling 搭配使用

Application Auto Scaling 與其他 AWS 服務整合，因此您可以新增擴展功能，以滿足應用程式的需求。自動擴展是服務的選擇性功能，在幾乎所有情況下都預設為停用。

下表列出可與 Application Auto Scaling 搭配使用 AWS 的服務，包括有關設定自動擴展的支援方法的資訊。您也可以對自訂資源使用 Application Auto Scaling。

- 主控台存取 — 您可以在目標服務的主控台設定擴展政策，將相容的 AWS 服務設定為開始自動擴展。
- CLI 存取 — 您可以使用 AWS CLI將相容的 AWS 服務設定為開始自動擴展。
- SDK 存取 – 您可以設定相容的 AWS 服務，以使用 AWS SDKs開始自動擴展。
- CloudFormation 存取 – 您可以設定相容的 AWS 服務，以使用 AWS CloudFormation 堆疊範本開始自動擴展。如需詳細資訊，請參閱使用 設定 Application Auto Scaling 資源 AWS CloudFormation。

AWS 服務	主控台存取1	CLI 存取	SDK 存取	CloudFormation 存取
AppStream 2.0	是	是	是	是
Aurora	是	是	是	是
Amazon Comprehend	否	是	是	是
Amazon DynamoDB	是	是	是	是

AWS 服務	主控台存取1	CLI 存取	SDK 存取	CloudFormation 存取
Amazon ECS	 是	 是	 是	 是
Amazon ElastiCache	 是	 是	 是	 是
Amazon EMR	 是	 是	 是	 是
Amazon Keyspaces	 是	 是	 是	 是
Lambda	 否	 是	 是	 是
Amazon MSK	 是	 是	 是	 是
Amazon Neptune	 否	 是	 是	 是
SageMaker AI	 是	 是	 是	 是
Spot 機群	 是	 是	 是	 是

AWS 服務	主控台存取1	CLI 存取	SDK 存取	CloudFormation 存取
WorkSpaces	 是	 是	 是	 是
自訂資源	 否	 是	 是	 是

1 用於設定擴展政策的主控台存取。大多數 服務不支援從主控台設定排程擴展。目前，只有 Amazon AppStream 2.0、ElastiCache 和 Spot Fleet 提供主控台存取以進行排程擴展。

Amazon AppStream 2.0 和 Application Auto Scaling

您可以使用目標追蹤擴展政策、步驟擴展政策和排程擴展，擴展 AppStream 2.0 機群。

使用下列資訊協助您將 AppStream 2.0 與 Application Auto Scaling 整合。

為 AppStream 2.0 建立的服務連結角色

將 AppStream 2.0 資源註冊為 Application Auto Scaling 可擴展的目標 AWS 帳戶 時，會在 中自動建立下列服務連結角色。此角色可讓 Application Auto Scaling 在您的帳戶內執行支援的操作。如需詳細資訊，請參閱Application Auto Scaling 的服務連結角色。

- `AWSServiceRoleForApplicationAutoScaling_AppStreamFleet`

服務連結角色所使用的服務委託人

上一節中的服務連結角色，只能由依據角色定義的信任關係所授權的服務委託人來擔任。Application Auto Scaling 使用的服務連結角色會將存取權授予下列服務委託人：

- `appstream.application-autoscaling.amazonaws.com`

向 Application Auto Scaling 將 AppStream 2.0 機群註冊為可擴展的目標

Application Auto Scaling 需要先有可擴展的目標，您才能為 AppStream 2.0 機群建立擴展政策或排定的動作。可擴展的目標是 Application Auto Scaling 可水平擴展和縮減的資源。可擴展的目標是由資源 ID、可擴展的維度和命名空間的組合來唯一識別。

如果您使用 AppStream 2.0 主控台設定自動擴展，則 AppStream 2.0 會自動為您註冊可擴展的目標。

如果您想要使用 CLI 或其中一個 AWS SDKs AWS 設定自動擴展，您可以使用下列選項：

- AWS CLI:

  為 AppStream 2.0 機群呼叫 register-scalable-target 命令。以下範例會替名為 `sample-fleet` 的機群註冊所需的容量，容量下限為 1 個機群執行個體，容量上限為 5 個機群執行個體。

  ```
  aws application-autoscaling register-scalable-target \
     --service-namespace appstream \
     --scalable-dimension appstream:fleet:DesiredCapacity \
     --resource-id fleet/sample-fleet \
     --min-capacity 1 \
     --max-capacity 5
  ```

  如果成功，此命令會傳回可擴展目標的 ARN。

  ```
  {
      "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-
  target/1234abcd56ab78cd901ef1234567890ab123"
  }
  ```

- AWS SDK：

  呼叫 RegisterScalableTarget 操作，並提供 `ResourceId`、`ScalableDimension`、`ServiceNamespace`、`MinCapacity` 及 `MaxCapacity` 作為參數。

相關資源

如需詳細資訊，請參閱《Amazon AppStream 2.0 管理指南》中的適用於 Amazon AppStream 2.0 的機群 Auto Scaling。 Amazon AppStream 2.0

Amazon Aurora 和 Application Auto Scaling

您可以使用目標追蹤擴展政策、步驟擴展政策和排程擴展，擴展 Aurora 資料庫叢集。

使用下列資訊協助您將 Aurora 與 Application Auto Scaling 整合。

為 Aurora 建立的服務連結角色

使用 Application Auto Scaling 將 Aurora 資源註冊為可擴展的目標 AWS 帳戶 時，系統會自動在 中建立下列服務連結角色。此角色可讓 Application Auto Scaling 在您的帳戶內執行支援的操作。如需詳細資訊，請參閱Application Auto Scaling 的服務連結角色。

- `AWSServiceRoleForApplicationAutoScaling_RDSCluster`

服務連結角色所使用的服務委託人

上一節中的服務連結角色，只能由依據角色定義的信任關係所授權的服務委託人來擔任。Application Auto Scaling 使用的服務連結角色會將存取權授予下列服務委託人：

- `rds.application-autoscaling.amazonaws.com`

向 Application Auto Scaling 將 Aurora 資料庫叢集註冊為可擴展的目標

Application Auto Scaling 需要先有可擴展的目標，您才能為 Aurora 叢集建立擴展政策或排定的動作。可擴展的目標是 Application Auto Scaling 可水平擴展和縮減的資源。可擴展的目標是由資源 ID、可擴展的維度和命名空間的組合來唯一識別。

如果您使用 Aurora 主控台設定自動擴展，則 Aurora 會自動為您註冊可擴展的目標。

如果您想要使用 CLI 或其中一個 AWS SDKs AWS 設定自動擴展，您可以使用下列選項：

- AWS CLI:

  為 Aurora 叢集呼叫 register-scalable-target 命令。以下範例會註冊名為 `my-db-cluster` 的叢集中的 Aurora 複本計數，容量下限為 1 個 Aurora 複本，容量上限為 8 個 Aurora 複本。

```
aws application-autoscaling register-scalable-target \
  --service-namespace rds \
  --scalable-dimension rds:cluster:ReadReplicaCount \
  --resource-id cluster:my-db-cluster \
```

```
  --min-capacity 1 \
  --max-capacity 8
```

如果成功，此命令會傳回可擴展目標的 ARN。

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
}
```

- AWS 開發套件：

  呼叫 RegisterScalableTarget 操作，並提供 `ResourceId`、`ScalableDimension`、`ServiceNamespace`、`MinCapacity` 及 `MaxCapacity` 作為參數。

相關資源

如需詳細資訊，請參閱《Amazon RDS Aurora 使用者指南》中的 Amazon Aurora Auto Scaling with Aurora 複本。

Amazon Comprehend 和 Application Auto Scaling

您可以使用目標追蹤擴展政策和排程擴展，擴展 Amazon Comprehend 文件分類和實體辨識器端點。

使用下列資訊協助您將 Amazon Comprehend 與 Application Auto Scaling 整合。

為 Amazon Comprehend 建立的服務連結角色

使用 Application Auto Scaling 將 Amazon Comprehend 資源註冊為可擴展的目標 AWS 帳戶 時，系統會自動在 中建立下列服務連結角色。此角色可讓 Application Auto Scaling 在您的帳戶內執行支援的操作。如需詳細資訊，請參閱Application Auto Scaling 的服務連結角色。

- `AWSServiceRoleForApplicationAutoScaling_ComprehendEndpoint`

服務連結角色所使用的服務委託人

上一節中的服務連結角色，只能由依據角色定義的信任關係所授權的服務委託人來擔任。Application Auto Scaling 使用的服務連結角色會將存取權授予下列服務委託人：

- comprehend.application-autoscaling.amazonaws.com

向 Application Auto Scaling 將 Amazon Comprehend 資源註冊為可擴展的目標

Application Auto Scaling 需要先有可擴展的目標，您才能為 Amazon Comprehend 文件分類或實體辨識器端點建立擴展政策或排定的動作。可擴展的目標是 Application Auto Scaling 可水平擴展和縮減的資源。可擴展的目標是由資源 ID、可擴展的維度和命名空間的組合來唯一識別。

若要使用 CLI 或其中一個 AWS SDKs AWS 設定自動擴展，您可以使用下列選項：

- AWS CLI:

  為文件分類端點呼叫 register-scalable-target 命令。以下範例會使用文件分類器端點的 ARN，註冊端點的模型所需使用的推論單位數，容量下限為 1 個推論單位，容量上限為 3 個推論單位。

```
aws application-autoscaling register-scalable-target \
  --service-namespace comprehend \
  --scalable-dimension comprehend:document-classifier-endpoint:DesiredInferenceUnits
 \
  --resource-id arn:aws:comprehend:us-west-2:123456789012:document-classifier-
endpoint/EXAMPLE \
  --min-capacity 1 \
  --max-capacity 3
```

  如果成功，此命令會傳回可擴展目標的 ARN。

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-
target/1234abcd56ab78cd901ef1234567890ab123"
}
```

  呼叫實體辨識器端點的 register-scalable-target 命令。以下範例會使用端點的 ARN，註冊實體辨識器的模型所需使用的推論單位數，容量下限為 1 個推論單位，容量上限為 3 個推論單位。

```
aws application-autoscaling register-scalable-target \
  --service-namespace comprehend \
  --scalable-dimension comprehend:entity-recognizer-endpoint:DesiredInferenceUnits \
  --resource-id arn:aws:comprehend:us-west-2:123456789012:entity-recognizer-
endpoint/EXAMPLE \
```

```
  --min-capacity 1 \
  --max-capacity 3
```

如果成功，此命令會傳回可擴展目標的 ARN。

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
}
```

- AWS 開發套件：

  呼叫 RegisterScalableTarget 操作，並提供 `ResourceId`、`ScalableDimension`、`ServiceNamespace`、`MinCapacity` 及 `MaxCapacity` 作為參數。

相關資源

如需詳細資訊，請參閱《Amazon Comprehend 開發人員指南》中的使用端點自動擴展。

Amazon DynamoDB 和 Application Auto Scaling

您可以使用目標追蹤擴展政策和排程擴展，擴展 DynamoDB 資料表和全域次要索引。

使用下列資訊協助您將 DynamoDB 與 Application Auto Scaling 整合。

為 DynamoDB 建立的服務連結角色

使用 Application Auto Scaling 將 DynamoDB 資源註冊為可擴展的目標 AWS 帳戶 時，系統會自動在中建立下列服務連結角色。此角色可讓 Application Auto Scaling 在您的帳戶內執行支援的操作。如需詳細資訊，請參閱Application Auto Scaling 的服務連結角色。

- `AWSServiceRoleForApplicationAutoScaling_DynamoDBTable`

服務連結角色所使用的服務委託人

上一節中的服務連結角色，只能由依據角色定義的信任關係所授權的服務委託人來擔任。Application Auto Scaling 使用的服務連結角色會將存取權授予下列服務委託人：

- dynamodb.application-autoscaling.amazonaws.com

向 Application Auto Scaling 將 DynamoDB 資源註冊為可擴展的目標

Application Auto Scaling 需要先有可擴展的目標，您才能為 DynamoDB 資料表或全域次要索引建立擴展政策或排定的動作。可擴展的目標是 Application Auto Scaling 可水平擴展和縮減的資源。可擴展的目標是由資源 ID、可擴展的維度和命名空間的組合來唯一識別。

如果您使用 DynamoDB 主控台設定自動擴展，則 DynamoDB 會自動為您註冊可擴展的目標。

如果您想要使用 CLI 或其中一個 AWS SDKs AWS 設定自動擴展，您可以使用下列選項：

- AWS CLI:

  為資料表的寫入容量呼叫 register-scalable-target 命令。下列範例會註冊名為 之資料表的佈建寫入容量my-table，最小容量為 5 個寫入容量單位，最大容量為 10 個寫入容量單位：

  ```
  aws application-autoscaling register-scalable-target \
    --service-namespace dynamodb \
    --scalable-dimension dynamodb:table:WriteCapacityUnits \
    --resource-id table/my-table \
    --min-capacity 5 \
    --max-capacity 10
  ```

  如果成功，此命令會傳回可擴展目標的 ARN：

  ```
  {
      "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
  }
  ```

  為資料表讀取容量呼叫 register-scalable-target 命令。下列範例會註冊名為 之資料表的佈建讀取容量my-table，最小容量為 5 個讀取容量單位，最大容量為 10 個讀取單位：

  ```
  aws application-autoscaling register-scalable-target \
    --service-namespace dynamodb \
    --scalable-dimension dynamodb:table:ReadCapacityUnits \
    --resource-id table/my-table \
    --min-capacity 5 \
    --max-capacity 10
  ```

如果成功，此命令會傳回可擴展目標的 ARN：

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
}
```

為全域次要索引的寫入容量呼叫 register-scalable-target 命令。下列範例會註冊名為 之全域次要索引的佈建寫入容量my-table-index，最小容量為 5 個寫入容量單位，最大容量為 10 個寫入容量單位：

```
aws application-autoscaling register-scalable-target \
  --service-namespace dynamodb \
  --scalable-dimension dynamodb:index:WriteCapacityUnits \
  --resource-id table/my-table/index/my-table-index \
  --min-capacity 5 \
  --max-capacity 10
```

如果成功，此命令會傳回可擴展目標的 ARN：

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
}
```

為全域次要索引的讀取容量呼叫 register-scalable-target 命令。下列範例會註冊名為 之全域次要索引的佈建讀取容量my-table-index，最小容量為 5 個讀取容量單位，最大容量為 10 個讀取容量單位：

```
aws application-autoscaling register-scalable-target \
  --service-namespace dynamodb \
  --scalable-dimension dynamodb:index:ReadCapacityUnits \
  --resource-id table/my-table/index/my-table-index \
  --min-capacity 5 \
  --max-capacity 10
```

如果成功，此命令會傳回可擴展目標的 ARN：

```
{
```

```
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-
target/1234abcd56ab78cd901ef1234567890ab123"
}
```

- AWS 開發套件：

  呼叫 RegisterScalableTarget 操作，並提供 `ResourceId`、`ScalableDimension`、`ServiceNamespace`、`MinCapacity` 及 `MaxCapacity` 作為參數。

相關資源

如果您才剛開始使用 Application Auto Scaling，您可以在下列文件中找到有關擴展 DynamoDB 資源的其他實用資訊：

- 《Amazon DynamoDB 開發人員指南》中的使用 DynamoDB Auto Scaling 管理輸送容量
- 《Amazon DynamoDB 開發人員指南》中的評估資料表的自動擴展設定
- 如何使用 AWS CloudFormation 在部落格上設定 DynamoDB 資料表和索引的自動擴展 AWS

Amazon ECS 和 Application Auto Scaling

您可以使用目標追蹤擴展政策、預測擴展政策、步進擴展政策和排程擴展來擴展 ECS 服務。

使用下列資訊協助您將 Amazon ECS 與 Application Auto Scaling 整合。

為 Amazon ECS 建立的服務連結角色

使用 Application Auto Scaling 將 Amazon ECS 資源註冊為可擴展的目標 AWS 帳戶 時，系統會自動在中建立下列服務連結角色。此角色可讓 Application Auto Scaling 在您的帳戶內執行支援的操作。如需詳細資訊，請參閱Application Auto Scaling 的服務連結角色。

- `AWSServiceRoleForApplicationAutoScaling_ECSService`

服務連結角色所使用的服務委託人

上一節中的服務連結角色，只能由依據角色定義的信任關係所授權的服務委託人來擔任。Application Auto Scaling 使用的服務連結角色會將存取權授予下列服務委託人：

- `ecs.application-autoscaling.amazonaws.com`

向 Application Auto Scaling 將 ECS 服務註冊為可擴展的目標

Application Auto Scaling 需要先有可擴展的目標，您才能為 Amazon ECS 服務建立擴展政策或排定的動作。可擴展的目標是 Application Auto Scaling 可水平擴展和縮減的資源。可擴展的目標是由資源 ID、可擴展的維度和命名空間的組合來唯一識別。

如果您使用 Amazon ECS 主控台設定自動擴展，則 Amazon ECS 會自動為您註冊可擴展的目標。

如果您想要使用 CLI 或其中一個 AWS SDKs AWS 設定自動擴展，您可以使用下列選項：

- AWS CLI:

  為 Amazon ECS 服務呼叫 register-scalable-target 命令。以下範例替 `default` 叢集上執行名為 `sample-app-service` 的服務註冊可擴展的目標，最小任務計數為一個任務，最大任務計數為 10 個任務。

  ```
  aws application-autoscaling register-scalable-target \
    --service-namespace ecs \
    --scalable-dimension ecs:service:DesiredCount \
    --resource-id service/default/sample-app-service \
    --min-capacity 1 \
    --max-capacity 10
  ```

  如果成功，此命令會傳回可擴展目標的 ARN。

  ```
  {
      "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
  }
  ```

- AWS SDK：

  呼叫 RegisterScalableTarget 操作，並提供 `ResourceId`、`ScalableDimension`、`ServiceNamespace`、`MinCapacity` 及 `MaxCapacity` 作為參數。

相關資源

如果您才剛開始使用 Application Auto Scaling，您可以在下列文件中找到有關擴展 Amazon ECS 資源的其他實用資訊：

- 《Amazon Elastic Container Service 開發人員指南》中的服務自動擴展
- 《Amazon Elastic Container Service 開發人員指南》中的最佳化 Amazon ECS 服務自動擴展

> **Note**
>
> 如需在 Amazon ECS 部署進行時暫停橫向擴展程序的說明，請參閱下列文件：《Amazon Elastic Container Service 開發人員指南》中的服務自動擴展和部署

ElastiCache 和 Application Auto Scaling

您可以使用目標追蹤擴展政策和排程擴展，水平擴展 Amazon ElastiCache 複寫群組 (Redis OSS 和 Valkey) 和 Memcached 自行設計的叢集。

若要將 ElastiCache 與 Application Auto Scaling 整合，請使用下列資訊。

為 ElastiCache 建立的服務連結角色

向 Application Auto Scaling 將 ElastiCache 資源註冊為可擴展的目標 AWS 帳戶 時，系統會自動在 中建立下列服務連結角色。此角色可讓 Application Auto Scaling 在您的帳戶內執行支援的操作。如需詳細資訊，請參閱Application Auto Scaling 的服務連結角色。

- `AWSServiceRoleForApplicationAutoScaling_ElastiCacheRG`

服務連結角色所使用的服務委託人

上一節中的服務連結角色，只能由依據角色定義的信任關係所授權的服務委託人來擔任。Application Auto Scaling 使用的服務連結角色會將存取權授予下列服務委託人：

- `elasticache.application-autoscaling.amazonaws.com`

使用 Application Auto Scaling 將 ElastiCache 資源註冊為可擴展的目標

Application Auto Scaling 需要可擴展的目標，才能為 ElastiCache 複寫群組、叢集或節點建立擴展政策或排程動作。可擴展的目標是 Application Auto Scaling 可水平擴展和縮減的資源。可擴展的目標是由資源 ID、可擴展的維度和命名空間的組合來唯一識別。

如果您使用 ElastiCache 主控台設定自動擴展，則 ElastiCache 會自動為您註冊可擴展的目標。

如果您想要使用 CLI 或其中一個 AWS SDKs AWS 設定自動擴展，您可以使用下列選項：

- AWS CLI:

  為 ElastiCache 複寫群組呼叫 register-scalable-target 命令。以下範例替名為 mycluster1 的複寫群組註冊所需的節點群組數量，容量下限為 1，容量上限為 5。

```
aws application-autoscaling register-scalable-target \
   --service-namespace elasticache \
   --scalable-dimension elasticache:replication-group:NodeGroups \
   --resource-id replication-group/mycluster1 \
   --min-capacity 1 \
   --max-capacity 5
```

  如果成功，此命令會傳回可擴展目標的 ARN。

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
}
```

  下列範例會為名為 的複寫群組，註冊每個節點群組所需的複本數量mycluster2，最小容量為 1，最大容量為 5。

```
aws application-autoscaling register-scalable-target \
   --service-namespace elasticache \
   --scalable-dimension elasticache:replication-group:Replicas \
   --resource-id replication-group/mycluster2 \
   --min-capacity 1 \
   --max-capacity 5
```

  如果成功，此命令會傳回可擴展目標的 ARN。

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-target/234abcd56ab78cd901ef1234567890ab1234"
}
```

下列範例會為名為 的叢集註冊所需的節點數量mynode1，容量下限為 20，容量上限為 50。

```
aws application-autoscaling register-scalable-target \
   --service-namespace elasticache \
   --scalable-dimension elasticache:cache-cluster:Nodes \
   --resource-id cache-cluster/mynode1 \
   --min-capacity 20 \
   --max-capacity 50
```

如果成功，此命令會傳回可擴展目標的 ARN。

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-target/01234abcd56ab78cd901ef1234567890ab12"
}
```

- AWS 開發套件：

  呼叫 RegisterScalableTarget 操作，並提供 `ResourceId`、`ScalableDimension`、`ServiceNamespace`、`MinCapacity` 及 `MaxCapacity` 作為參數。

相關資源

如需詳細資訊，請參閱《Amazon ElastiCache 使用者指南》中的 Auto Scaling Valkey 和 Redis OSS 叢集和 Memcached 的 Scaling 叢集。

Amazon Keyspaces (適用於 Apache Cassandra) 和 Application Auto Scaling

您可以使用目標追蹤擴展政策和排程擴展來擴展 Amazon Keyspaces 資料表。

使用下列資訊協助您將 Amazon Keyspaces 與 Application Auto Scaling 整合。

為 Amazon Keyspaces 建立的服務連結角色

使用 Application Auto Scaling 將 Amazon Keyspaces 資源註冊為可擴展的目標 AWS 帳戶 時，系統會自動在 中建立下列服務連結角色。此角色可讓 Application Auto Scaling 在您的帳戶內執行支援的操作。如需詳細資訊，請參閱Application Auto Scaling 的服務連結角色。

- AWSServiceRoleForApplicationAutoScaling_CassandraTable

服務連結角色所使用的服務委託人

上一節中的服務連結角色，只能由依據角色定義的信任關係所授權的服務委託人來擔任。Application Auto Scaling 使用的服務連結角色會將存取權授予下列服務委託人：

- cassandra.application-autoscaling.amazonaws.com

向 Application Auto Scaling 將 Amazon Keyspaces 資料表註冊為可擴展的目標

Application Auto Scaling 需要先有可擴展的目標，您才能為 Amazon Keyspaces 資料表建立擴展政策或排定的動作。可擴展的目標是 Application Auto Scaling 可水平擴展和縮減的資源。可擴展的目標是由資源 ID、可擴展的維度和命名空間的組合來唯一識別。

如果您使用 Amazon Keyspaces 主控台設定自動擴展，則 Amazon Keyspaces 會自動為您註冊可擴展的目標。

如果您想要使用 CLI 或其中一個 AWS SDKs AWS 設定自動擴展，您可以使用下列選項：

- AWS CLI:

  為 Amazon Keyspaces 資料表呼叫 register-scalable-target 命令。以下範例替名為 mytable 的資料表註冊佈建寫入容量，容量下限為 5 個寫入容量單位，容量上限為 10 個寫入容量單位。

```
aws application-autoscaling register-scalable-target \
  --service-namespace cassandra \
  --scalable-dimension cassandra:table:WriteCapacityUnits \
  --resource-id keyspace/mykeyspace/table/mytable \
  --min-capacity 5 \
  --max-capacity 10
```

  如果成功，此命令會傳回可擴展目標的 ARN。

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
}
```

以下範例替名為 mytable 的資料表註冊佈建讀取容量，容量下限為 5 個讀取容量單位，容量上限為 10 個讀取容量單位。

```
aws application-autoscaling register-scalable-target \
  --service-namespace cassandra \
  --scalable-dimension cassandra:table:ReadCapacityUnits \
  --resource-id keyspace/mykeyspace/table/mytable \
  --min-capacity 5 \
  --max-capacity 10
```

如果成功，此命令會傳回可擴展目標的 ARN。

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
}
```

- AWS 開發套件：

  呼叫 RegisterScalableTarget 操作，並提供 ResourceId、ScalableDimension、ServiceNamespace、MinCapacity 及 MaxCapacity 作為參數。

相關資源

如需詳細資訊，請參閱《Amazon Keyspaces 開發人員指南》中的使用 Amazon Keyspaces 自動擴展自動管理輸送量容量。

AWS Lambda 和 Application Auto Scaling

您可以使用目標追蹤擴展政策和排程擴展來擴展 AWS Lambda 佈建並行。

使用下列資訊協助您將 Lambda 與 Application Auto Scaling 整合。

為 Lambda 建立的服務連結角色

向 Application Auto Scaling 將 Lambda 資源註冊為可擴展的目標 AWS 帳戶 時，系統會自動在 中建立下列服務連結角色。此角色可讓 Application Auto Scaling 在您的帳戶內執行支援的操作。如需詳細資訊，請參閱Application Auto Scaling 的服務連結角色。

- `AWSServiceRoleForApplicationAutoScaling_LambdaConcurrency`

服務連結角色所使用的服務委託人

上一節中的服務連結角色，只能由依據角色定義的信任關係所授權的服務委託人來擔任。Application Auto Scaling 使用的服務連結角色會將存取權授予下列服務委託人：

- `lambda.application-autoscaling.amazonaws.com`

向 Application Auto Scaling 將 Lambda 函數註冊為可擴展的目標

Application Auto Scaling 需要先有可擴展的目標，您才能為 Lambda 函數建立擴展政策或排定的動作。可擴展的目標是 Application Auto Scaling 可水平擴展和縮減的資源。可擴展的目標是由資源 ID、可擴展的維度和命名空間的組合來唯一識別。

若要使用 CLI 或其中一個 AWS SDKs AWS 設定自動擴展，您可以使用下列選項：

- AWS CLI:

  為 Lambda 函數呼叫 register-scalable-target 命令。以下範例替名為 `my-function` 的函數註冊別名為 BLUE 的佈建並行，容量下限為 0，容量上限為 100。

  ```
  aws application-autoscaling register-scalable-target \
    --service-namespace lambda \
    --scalable-dimension lambda:function:ProvisionedConcurrency \
    --resource-id function:my-function:BLUE \
    --min-capacity 0 \
    --max-capacity 100
  ```

  如果成功，此命令會傳回可擴展目標的 ARN。

  ```
  {
      "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
  }
  ```

- AWS SDK：

呼叫 RegisterScalableTarget 操作，並提供 `ResourceId`、`ScalableDimension`、`ServiceNamespace`、`MinCapacity` 及 `MaxCapacity` 作為參數。

相關資源

如果您才剛開始使用 Application Auto Scaling，您可以在下列文件中找到有關擴展 Lambda 函數的其他實用資訊：

- 《 AWS Lambda 開發人員指南》中的設定佈建並行
- 在 AWS 部落格上為週期性尖峰用量排程 Lambda 佈建並行

Amazon Managed Streaming for Apache Kafka (MSK) 和 Application Auto Scaling

您可以使用目標追蹤擴展政策來擴展 Amazon MSK 叢集儲存。透過目標追蹤政策進行縮減的功能已停用。

使用下列資訊協助您將 Amazon MSK 與 Application Auto Scaling 整合。

為 Amazon MSK 建立的服務連結角色

使用 Application Auto Scaling 將 Amazon MSK 資源註冊為可擴展的目標 AWS 帳戶 時，系統會自動在 中建立下列服務連結角色。此角色可讓 Application Auto Scaling 在您的帳戶內執行支援的操作。如需詳細資訊，請參閱Application Auto Scaling 的服務連結角色。

- `AWSServiceRoleForApplicationAutoScaling_KafkaCluster`

服務連結角色所使用的服務委託人

上一節中的服務連結角色，只能由依據角色定義的信任關係所授權的服務委託人來擔任。Application Auto Scaling 使用的服務連結角色會將存取權授予下列服務委託人：

- `kafka.application-autoscaling.amazonaws.com`

向 Application Auto Scaling 將 Amazon MSK 叢集儲存註冊為可擴展的目標

Application Auto Scaling 需要先有可擴展的目標，您才能為 Amazon MSK 叢集的每一代理程式的儲存磁碟區大小建立擴展政策。可擴展的目標是指 Application Auto Scaling 可擴展或縮減的資源。可擴展的目標是由資源 ID、可擴展的維度和命名空間的組合來唯一識別。

如果您使用 Amazon MSK 主控台設定自動擴展，則 Amazon MSK 會自動為您註冊可擴展的目標。

如果您想要使用 CLI 或其中一個 AWS SDKs AWS 設定自動擴展，您可以使用下列選項：

- AWS CLI:

  為 Amazon MSK 叢集呼叫 register-scalable-target 命令。以下範例會為 Amazon MSK 叢集註冊每一代理程式的儲存磁碟區大小，容量下限為 100 GiB，容量上限為 800 GiB。

  ```
  aws application-autoscaling register-scalable-target \
     --service-namespace kafka \
     --scalable-dimension kafka:broker-storage:VolumeSize \
     --resource-id arn:aws:kafka:us-east-1:123456789012:cluster/demo-
  cluster-1/6357e0b2-0e6a-4b86-a0b4-70df934c2e31-5 \
     --min-capacity 100 \
     --max-capacity 800
  ```

  如果成功，此命令會傳回可擴展目標的 ARN。

  ```
  {
      "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-
  target/1234abcd56ab78cd901ef1234567890ab123"
  }
  ```

- AWS SDK：

  呼叫 RegisterScalableTarget 操作，並提供 `ResourceId`、`ScalableDimension`、`ServiceNamespace`、`MinCapacity` 及 `MaxCapacity` 作為參數。

> **Note**
>
> 當 Amazon MSK 叢集是可擴展的目標時，縮減會停用且無法啟用。

相關資源

如需詳細資訊，請參閱《Amazon Managed Streaming for Apache Kafka 開發人員指南》中的 Amazon MSK 叢集的自動擴展。

Amazon Neptune 和 Application Auto Scaling

您可以使用目標追蹤擴展政策和排程擴展來擴展 Neptune 叢集。

使用下列資訊協助您將 Neptune 與 Application Auto Scaling 整合。

為 Neptune 建立的服務連結角色

向 Application Auto Scaling 將 Neptune 資源註冊為可擴展的目標 AWS 帳戶 時，系統會自動在 中建立下列服務連結角色。此角色可讓 Application Auto Scaling 在您的帳戶內執行支援的操作。如需詳細資訊，請參閱Application Auto Scaling 的服務連結角色。

- `AWSServiceRoleForApplicationAutoScaling_NeptuneCluster`

服務連結角色所使用的服務委託人

上一節中的服務連結角色，只能由依據角色定義的信任關係所授權的服務委託人來擔任。Application Auto Scaling 使用的服務連結角色會將存取權授予下列服務委託人：

- `neptune.application-autoscaling.amazonaws.com`

在 Application Auto Scaling 中將 Neptune 叢集註冊為可擴展的目標

Application Auto Scaling 需要先有可擴展的目標，您才能為 Neptune 叢集建立擴展政策或排定的動作。可擴展的目標是 Application Auto Scaling 可水平擴展和縮減的資源。可擴展的目標是由資源 ID、可擴展的維度和命名空間的組合來唯一識別。

若要使用 CLI 或其中一個 AWS SDKs AWS 設定自動擴展，您可以使用下列選項：

- AWS CLI:

  為 Neptune 叢集呼叫 register-scalable-target 命令。以下範例會替名為 `mycluster` 的叢集註冊所需的容量，容量下限為 1，上限為 8。

```
aws application-autoscaling register-scalable-target \
  --service-namespace neptune \
  --scalable-dimension neptune:cluster:ReadReplicaCount \
  --resource-id cluster:mycluster \
  --min-capacity 1 \
  --max-capacity 8
```

如果成功，此命令會傳回可擴展目標的 ARN。

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
}
```

- AWS SDK：

  呼叫 RegisterScalableTarget 操作，並提供 `ResourceId`、`ScalableDimension`、`ServiceNamespace`、`MinCapacity` 及 `MaxCapacity` 作為參數。

相關資源

如需詳細資訊，請參閱《Neptune 使用者指南》中的自動擴展 Amazon Neptune 資料庫叢集中的複本數量。

Amazon SageMaker AI 和 Application Auto Scaling

您可以使用目標追蹤擴展政策、步進擴展政策和排程擴展，來擴展 SageMaker AI 端點變體、無伺服器端點的佈建並行，以及推論元件。

使用以下資訊來協助您整合 SageMaker AI 與 Application Auto Scaling。

為 SageMaker AI 建立的服務連結角色

將 SageMaker AI 資源註冊為 Application Auto Scaling 可擴展的目標 AWS 帳戶 時，系統會自動在 中建立下列服務連結角色。此角色可讓 Application Auto Scaling 在您的帳戶內執行支援的操作。如需詳細資訊，請參閱Application Auto Scaling 的服務連結角色。

- `AWSServiceRoleForApplicationAutoScaling_SageMakerEndpoint`

服務連結角色所使用的服務委託人

上一節中的服務連結角色，只能由依據角色定義的信任關係所授權的服務委託人來擔任。Application Auto Scaling 使用的服務連結角色會將存取權授予下列服務委託人：

- `sagemaker.application-autoscaling.amazonaws.com`

使用 Application Auto Scaling 將 SageMaker AI 端點變體註冊為可擴展的目標

Application Auto Scaling 需要可擴展的目標，才能為 SageMaker AI 模型（變體）建立擴展政策或排程動作。可擴展的目標是 Application Auto Scaling 可水平擴展和縮減的資源。可擴展的目標是由資源 ID、可擴展的維度和命名空間的組合來唯一識別。

如果您使用 SageMaker AI 主控台設定自動擴展，則 SageMaker AI 會自動為您註冊可擴展的目標。

如果您想要使用 CLI 或其中一個 AWS SDKs AWS 設定自動擴展，您可以使用下列選項：

- AWS CLI:

  為產品變體呼叫 register-scalable-target 命令。以下範例會為 `my-endpoint` 端點上執行名為 `my-variant` 的產品變體註冊所需的執行個體計數，容量下限為 1 個執行個體，容量上限為 8 個執行個體。

  ```
  aws application-autoscaling register-scalable-target \
    --service-namespace sagemaker \
    --scalable-dimension sagemaker:variant:DesiredInstanceCount \
    --resource-id endpoint/my-endpoint/variant/my-variant \
    --min-capacity 1 \
    --max-capacity 8
  ```

  如果成功，此命令會傳回可擴展目標的 ARN。

  ```
  {
      "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
  }
  ```

- AWS 開發套件：

呼叫 RegisterScalableTarget 操作，並提供 `ResourceId`、`ScalableDimension`、`ServiceNamespace`、`MinCapacity` 及 `MaxCapacity` 作為參數。

向 Application Auto Scaling 將無伺服器端點的佈建並行註冊為可擴展的目標

Application Auto Scaling 還需要先有可擴展的目標，您才能為無伺服器端點的佈建並行建立擴展政策或排定的動作。

如果您使用 SageMaker AI 主控台設定自動擴展，則 SageMaker AI 會自動為您註冊可擴展的目標。

否則，使用下列其中一種方法來註冊可擴展的目標：

- AWS CLI:

  為產品變體呼叫 register-scalable-target 命令。以下範例會為 `my-endpoint` 端點上執行名為 `my-variant` 的產品變體註冊佈建並行，容量下限為一，容量上限為十。

  ```
  aws application-autoscaling register-scalable-target \
    --service-namespace sagemaker \
    --scalable-dimension sagemaker:variant:DesiredProvisionedConcurrency \
    --resource-id endpoint/my-endpoint/variant/my-variant \
    --min-capacity 1 \
    --max-capacity 10
  ```

  如果成功，此命令會傳回可擴展目標的 ARN。

  ```
  {
      "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
  }
  ```

- AWS 開發套件：

  呼叫 RegisterScalableTarget 操作，並提供 `ResourceId`、`ScalableDimension`、`ServiceNamespace`、`MinCapacity` 及 `MaxCapacity` 作為參數。

在 Application Auto Scaling 中將推論元件註冊為可擴展的目標

Application Auto Scaling 需要先有可擴展的目標，您才能為推論元件建立擴展政策或排定的動作。

- AWS CLI:

  為推論元件呼叫 register-scalable-target 命令。下列範例會替名為的推論元件註冊所需的複本計數量`my-inference-component`，容量下限為 0，上限為 3。

```
aws application-autoscaling register-scalable-target \
  --service-namespace sagemaker \
  --scalable-dimension sagemaker:inference-component:DesiredCopyCount \
  --resource-id inference-component/my-inference-component \
  --min-capacity 0 \
  --max-capacity 3
```

  如果成功，此命令會傳回可擴展目標的 ARN。

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
}
```

- AWS 開發套件：

  呼叫 RegisterScalableTarget 操作，並提供 `ResourceId`、`ScalableDimension`、`ServiceNamespace`、`MinCapacity` 及 `MaxCapacity` 作為參數。

相關資源

如果您才剛開始使用 Application Auto Scaling，您可以在 Amazon SageMaker AI 開發人員指南中找到有關擴展 SageMaker AI 資源的其他實用資訊： Amazon SageMaker

- 自動擴展 Amazon SageMaker AI 模型
- 自動擴展無伺服器端點的佈建並行
- 設定多模型端點部署的自動擴展政策
- 自動擴展非同步端點

> **Note**
>
> 在 2023 年，SageMaker AI 推出了以即時推論端點為基礎的新推論功能。您可以使用端點組態來建立 SageMaker AI 端點，該組態會定義端點的執行個體類型和初始執行個體計數。然後，建立推論元件，這是 SageMaker AI 託管物件，可用來將模型部署到端點。如需有關擴展推論元件的資訊，請參閱 AWS 部落格上的 Amazon SageMaker AI 新增了新的推論功能，以協助降低基礎模型部署成本和延遲，以及使用 Amazon SageMaker AI 的最新功能平均降低模型部署成本 50%。

Amazon EC2 Spot 機群和 Application Auto Scaling

您可以使用目標追蹤擴展政策、步驟擴展政策和排程擴展來擴展 Spot 機群。

使用下列資訊協助您將 Spot 機群 與 Application Auto Scaling 整合。

為 Spot 機群建立的服務連結角色

向 Application Auto Scaling 將 Spot Fleet 資源註冊為可擴展的目標 AWS 帳戶 時，系統會自動在 中建立下列服務連結角色。此角色可讓 Application Auto Scaling 在您的帳戶內執行支援的操作。如需詳細資訊，請參閱Application Auto Scaling 的服務連結角色。

- `AWSServiceRoleForApplicationAutoScaling_EC2SpotFleetRequest`

服務連結角色所使用的服務委託人

上一節中的服務連結角色，只能由依據角色定義的信任關係所授權的服務委託人來擔任。Application Auto Scaling 使用的服務連結角色會將存取權授予下列服務委託人：

- `ec2.application-autoscaling.amazonaws.com`

向 Application Auto Scaling 將 Spot 機群註冊為可擴展的目標

Application Auto Scaling 需要先有可擴展的目標，您才能為 Spot 機群建立擴展政策或排定的動作。可擴展的目標是 Application Auto Scaling 可水平擴展和縮減的資源。可擴展的目標是由資源 ID、可擴展的維度和命名空間的組合來唯一識別。

如果您使用 Spot 機群主控台設定自動擴展，則 Spot 機群會自動為您註冊可擴展的目標。

如果您想要使用 CLI 或其中一個 AWS SDKs AWS 設定自動擴展，您可以使用下列選項：

- AWS CLI:

  為 Spot 機群呼叫 register-scalable-target 命令。以下範例會使用請求 ID 註冊 Spot 機群的目標容量，容量下限為 2 個執行個體，容量上限為 10 個執行個體。

```
aws application-autoscaling register-scalable-target \
  --service-namespace ec2 \
  --scalable-dimension ec2:spot-fleet-request:TargetCapacity \
  --resource-id spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE \
  --min-capacity 2 \
  --max-capacity 10
```

  如果成功，此命令會傳回可擴展目標的 ARN。

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
}
```

- AWS SDK：

  呼叫 RegisterScalableTarget 操作，並提供 `ResourceId`、`ScalableDimension`、`ServiceNamespace`、`MinCapacity` 及 `MaxCapacity` 作為參數。

相關資源

如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的了解 Spot 機群的自動擴展。

Amazon WorkSpaces 和 Application Auto Scaling

您可以使用目標追蹤擴展政策、步驟擴展政策和排程擴展來擴展 WorkSpaces 集區。

使用以下資訊來協助您整合 WorkSpaces 與 Application Auto Scaling。

為 WorkSpaces 建立的服務連結角色

當您向 Application Auto Scaling 將 WorkSpaces 資源註冊為可擴展的目標 AWS 帳戶 時，Application Auto Scaling 會自動在 AWSServiceRoleForApplicationAutoScaling_WorkSpacesPool中建立名為 的服務連結角色。如需詳細資訊，請參閱Application Auto Scaling 的服務連結角色。

此服務連結角色使用受管政策 AWSApplicationAutoscalingWorkSpacesPoolPolicy。此政策授予 Application Auto Scaling 許可，以代表您呼叫 Amazon WorkSpaces。如需詳細資訊，請參閱《 AWS 受管政策參考》中的 AWSApplicationAutoscalingWorkSpacesPoolPolicy。

服務連結角色所使用的服務委託人

服務連結角色信任下列服務主體擔任該角色：

- `workspaces.application-autoscaling.amazonaws.com`

使用 Application Auto Scaling 將 WorkSpaces 集區註冊為可擴展的目標

Application Auto Scaling 需要可擴展的目標，才能為 WorkSpaces 建立擴展政策或排程動作。可擴展的目標是 Application Auto Scaling 可水平擴展和縮減的資源。可擴展的目標是由資源 ID、可擴展的維度和命名空間的組合來唯一識別。

如果您使用 WorkSpaces 主控台設定自動擴展，則 WorkSpaces 會自動為您註冊可擴展的目標。

如果您想要使用 CLI 或其中一個 AWS SDKs AWS 設定自動擴展，您可以使用下列選項：

- AWS CLI:

  呼叫 WorkSpaces 集區的 register-scalable-target 命令。下列範例會使用其請求 ID 註冊 WorkSpaces 集區的目標容量，最小容量為兩個虛擬桌面，最大容量為十個虛擬桌面。

```
aws application-autoscaling register-scalable-target \
  --service-namespace workspaces \
  --resource-id workspacespool/wspool-abcdef012 \
  --scalable-dimension workspaces:workspacespool:DesiredUserSessions \
  --min-capacity 2 \
  --max-capacity 10
```

  如果成功，此命令會傳回可擴展目標的 ARN。

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-
target/1234abcd56ab78cd901ef1234567890ab123"
}
```

- AWS 開發套件：

  呼叫 RegisterScalableTarget 操作，並提供 `ResourceId`、`ScalableDimension`、`ServiceNamespace`、`MinCapacity` 及 `MaxCapacity` 作為參數。

相關資源

如需詳細資訊，請參閱《Amazon WorkSpaces 管理指南》中的適用於 WorkSpaces 集區的 Auto Scaling。 Amazon WorkSpaces

自訂資源和 Application Auto Scaling

您可以使用目標追蹤擴展政策、步驟擴展政策和排程擴展來擴展自訂資源。

使用下列資訊協助您將自訂資源與 Application Auto Scaling 整合。

為自訂資源建立的服務連結角色

將自訂資源註冊為 Application Auto Scaling 可擴展的目標 AWS 帳戶 時，會在 中自動建立下列服務連結角色。此角色可讓 Application Auto Scaling 在您的帳戶內執行支援的操作。如需詳細資訊，請參閱Application Auto Scaling 的服務連結角色。

- `AWSServiceRoleForApplicationAutoScaling_CustomResource`

服務連結角色所使用的服務委託人

上一節中的服務連結角色，只能由依據角色定義的信任關係所授權的服務委託人來擔任。Application Auto Scaling 使用的服務連結角色會將存取權授予下列服務委託人：

- `custom-resource.application-autoscaling.amazonaws.com`

向 Application Auto Scaling 將自訂資源註冊為可擴展的目標

Application Auto Scaling 需要先有可擴展的目標，您才能為自訂資源建立擴展政策或排定的動作。可擴展的目標是 Application Auto Scaling 可水平擴展和縮減的資源。可擴展的目標是由資源 ID、可擴展的維度和命名空間的組合來唯一識別。

若要使用 CLI 或其中一個 AWS SDKs AWS 設定自動擴展，您可以使用下列選項：

- AWS CLI:

  為自訂資源呼叫 register-scalable-target 命令。以下範例會將自訂資源註冊為可擴展的目標，所需計數下限為 1 個容量單位，所需計數上限為 10 個容量單位。`custom-resource-id.txt` 檔案包含資源 ID 的識別字串，代表自訂資源通過 Amazon API Gateway 端點的路徑。

  ```
  aws application-autoscaling register-scalable-target \
    --service-namespace custom-resource \
    --scalable-dimension custom-resource:ResourceType:Property \
    --resource-id file://~/custom-resource-id.txt \
    --min-capacity 1 \
    --max-capacity 10
  ```

  `custom-resource-id.txt` 的內容：

  ```
  https://example.execute-api.us-west-2.amazonaws.com/prod/
  scalableTargetDimensions/1-23456789
  ```

  如果成功，此命令會傳回可擴展目標的 ARN。

  ```
  {
      "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-
  target/1234abcd56ab78cd901ef1234567890ab123"
  }
  ```

- AWS SDK：

  呼叫 RegisterScalableTarget 操作，並提供 `ResourceId`、`ScalableDimension`、`ServiceNamespace`、`MinCapacity` 及 `MaxCapacity` 作為參數。

相關資源

如果您才剛開始使用 Application Auto Scaling，您可以在下列文件中找到有關擴展自訂資源的其他實用資訊：

GitHub 儲存庫

使用 設定 Application Auto Scaling 資源 AWS CloudFormation

Application Auto Scaling 已與 整合 AWS CloudFormation，這項服務可協助您建立和設定 AWS 資源的模型，以減少建立和管理資源和基礎設施的時間。您可以建立範本來描述您想要的所有 AWS 資源，並為您 AWS CloudFormation 佈建和設定這些資源。

使用 時 AWS CloudFormation，您可以重複使用範本，以一致且重複地設定 Application Auto Scaling 資源。描述您的資源一次，然後在多個 AWS 帳戶 和 區域中逐一佈建相同的資源。

Application Auto Scaling 和 AWS CloudFormation 範本

若要為 Application Auto Scaling 及相關服務佈建和設定資源，您必須了解 AWS CloudFormation 範本。範本是以 JSON 或 YAML 格式化的文本檔案。這些範本說明您要在 AWS CloudFormation 堆疊中佈建的資源。如果您不熟悉 JSON 或 YAML，您可以使用 AWS CloudFormation 設計工具來協助您開始使用 AWS CloudFormation 範本。如需更多詳細資訊，請參閱 AWS CloudFormation 使用者指南 中的 什麼是 AWS CloudFormation 設計器？。

當您為 Application Auto Scaling 資源建立堆疊範本時，必須提供下列項目：

- 目標服務的命名空間 (例如，**appstream**)。請參閱 AWS::ApplicationAutoScaling::ScalableTarget 參考，以取得服務命名空間。
- 與目標資源相關聯的可擴展維度 (例如，**appstream:fleet:DesiredCapacity**)。請參閱 AWS::ApplicationAutoScaling::ScalableTarget 參考，以取得可擴展的維度。
- 目標資源的資源 ID (例如，**fleet/sample-fleet**)。請參閱 AWS::ApplicationAutoScaling::ScalableTarget 參考，以取得特定資源 ID 的語法和範例的相關資訊。
- 目標資源的服務連結角色 (例如，**arn:aws:iam::012345678910:role/aws-service-role/appstream.application-autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_AppStreamFleet**)。請參閱服務連結角色 ARN 參考表，以取得角色 ARN。

若要進一步了解 Application Auto Scaling 資源，請參閱《AWS CloudFormation 使用者指南》中的 Application Auto Scaling 參考資料。

範本程式碼片段範例

您可以在 AWS CloudFormation 使用者指南的下列章節 AWS CloudFormation 中找到範本中包含的範例程式碼片段：

- 如需擴展政策和排程動作的範例，請參閱使用 設定 Application Auto Scaling 資源 AWS CloudFormation。
- 如需擴展政策的更多範例，請參閱 AWS::ApplicationAutoScaling::ScalingPolicy。

進一步了解 AWS CloudFormation

若要進一步了解 AWS CloudFormation，請參閱下列資源：

- AWS CloudFormation
- AWS CloudFormation 使用者指南
- AWS CloudFormation API 參考
- AWS CloudFormation 命令列介面使用者指南

Application Auto Scaling 的排程擴展

透過排程擴展，您可以建立排程動作，根據所預測的負載變動，在特定時間增加或減少容量，自動調整應用程式規模。如此一來，您便能主動調整應用程式規模，以符合負載變動預測。

例如，假設您每週的流量模式是週間流量增加，越靠近週末流量越少，您可以到 Application Auto Scaling 設定與此模式一致的擴展排程：

- 透過排程動作，在星期三早上提高先前為可擴展目標設定的最低容量，藉此增加容量。
- 到了星期五晚上，再透過另一個排程動作，降低先前為可擴充目標設定的最高容量，以減少容量。

藉由這些排定的擴展動作，您可將費用和效能調整到最佳狀態。您的應用程式可以有足夠的容量處理週間的流量高峰，但在其他時段不必過度佈建不需要的容量。

您可以搭配利用排程擴展和擴展政策，以主動和被動的方式處理規模擴展作業，同時享有這兩種方法的好處。執行排定的擴展動作後，擴展政策可以繼續決定是否進一步擴展容量。這有助於確保您有足夠的容量來處理應用程式的負載。當應用程式擴展以滿足需求時，目前的容量必須落在您排定的動作所設定的容量上下限之內。

目錄

Application Auto Scaling 的排程擴展如何運作

本主題說明排程擴展的運作方式，並介紹有效使用它所需的重要考量。

目錄

- 建立、管理及刪除排定動作常用的命令
- 相關資源
- 限制

運作方式

若要使用排程擴展，請建立排定的動作，以告知 Application Auto Scaling 在特定的時間執行擴展活動。建立排定的動作時，您需要指定可擴展的目標、何時進行擴展活動、容量下限和容量上限。您可以建立僅擴展一次或依週期性排程擴展的排程動作。

在指定的時間，Application Auto Scaling 會將目前容量與指定的容量上下限相比較，以根據新的容量值來擴展。

- 如果目前的容量低於指定的容量下限，則 Application Auto Scaling 會水平擴展 (增加容量) 到指定的容量下限。
- 如果目前的容量高於指定的容量上限，則 Application Auto Scaling 會縮減 (減少容量) 到指定的容量上限。

考量事項

當您建立排程動作時，請謹記下列事項：

- 排定的動作會在指定的日期和時間將 `MinCapacity` 和 `MaxCapacity` 設為排定動作所指定的值。請求可以選擇僅包含這些大小之一。例如，您可以建立僅指定最小容量的排定動作。然而，在某些情況下，您必須包含這兩種大小，確保新的最小容量不會大於最大容量，或者新的最大容量不會小於最小容量。
- 依預設，您設定的週期性排程會使用國際標準時間 (UTC)。您可以變更時區以對應至您當地的時區或網路另一個部分的時區。如果您指定的時區遵守日光節約時間，則動作會依據日光節約時間 (DST) 自動調整。如需詳細資訊，請參閱使用 Application Auto Scaling 排程週期性擴展動作。
- 您可以對可擴展的目標暫時停用排定的擴展。這可協助您避免排程動作處於作用中狀態，而不需要將其刪除。然後，您可以在想要再次使用時繼續執行排程擴展。如需詳細資訊，請參閱Application Auto Scaling 暫停和繼續擴展。
- 對於同一可擴展的目標，會保證排定動作的執行順序，但對於跨可擴展目標的排定動作則無法保證。
- 若要成功完成排定的動作，指定的資源在目標服務中必須處於可擴展狀態。如果不是，則該請求會失敗並傳回一條錯誤訊息，例如 `Resource Id [ActualResourceId] is not`

`scalable. Reason: The status of all DB instances must be 'available' or 'incompatible-parameters'`。

- 由於 Application Auto Scaling 和目標服務的分散式特性，從觸發排定的動作到目標服務履行擴展動作之間，可能會延遲幾秒鐘。由於排定的動作是按照指定的順序執行，如果排定的動作彼此的開始時間很接近，就可能執行越久。

建立、管理及刪除排定動作常用的命令

常用於排程擴展的命令包括：

- register-scalable-target 將 AWS 或自訂資源註冊為可擴展的目標 (Application Auto Scaling 可以擴展的資源），以及暫停和繼續擴展。
- put-scheduled-action，新增或修改現有可擴展目標的排定動作。
- describe-scaling-activities 傳回 AWS 區域中擴展活動的相關資訊。
- describe-scheduled-actions 傳回 AWS 區域中排程動作的相關資訊。
- delete-scheduled-action，刪除排定的動作。

相關資源

如需使用排程擴展的詳細範例，請參閱 AWS 運算部落格上的部落格文章排程 AWS Lambda 佈建並行以取得週期性尖峰用量。

如需有關建立 Auto Scaling 群組的排程動作之詳細資訊，請參閱《Amazon EC2 Auto Scaling 使用者指南》中的「Auto Scaling 群組的排程擴展」。

限制

以下是使用排程擴展時的限制：

- 每個可擴展目標的排定動作名稱必須是唯一名稱。
- Application Auto Scaling 在排程表達式中不提供第二層精確度。使用 cron 表達式的最佳解析是一分鐘。
- 可擴展的目標不能是 Amazon MSK 叢集。Amazon MSK 不支援排程擴展。
- 主控台對可擴展資源的檢視、新增、更新或移除排程動作的存取，視您所使用的資源而定。如需詳細資訊，請參閱AWS 服務 可與 Application Auto Scaling 搭配使用。

使用 建立 Application Auto Scaling 的排程動作 AWS CLI

下列範例示範如何使用 AWS CLI put-scheduled-action 命令建立排程動作。指定新的容量時，可指定最低容量、最高容量或一併指定最低和最高容量。

這些範例針對與 Application Auto Scaling 整合的幾個服務使用可擴展的目標。若要使用不同的可擴展目標，請在 中指定其命名空間`--service-namespace`、在 中指定其可擴展維度`--scalable-dimension`，以及在 中指定其資源 ID`--resource-id`。

使用 時 AWS CLI，請記住，您的命令會在為設定檔 AWS 區域 設定的 中執行。如果您想在不同區域中執行命令，則可變更設定檔的預設區域，或搭配 `--region` 參數使用命令。

範例

- 建立只發生一次的排程動作
- 建立依週期性間隔執行的排定動作
- 建立依週期性排程執行的排程動作
- 建立一次性排定動作並指定時區
- 建立指定時區的週期性排程動作

建立只發生一次的排程動作

若只要在指定的日期和時間自動擴展可擴展的目標一次，請使用 `--schedule "at(`*`yyyy-mm-ddThh:mm:ss`*`)"` 選項。

Example 範例：僅擴增一次

以下是建立排定動作在特定日期和時間水平擴展容量的範例。

在 `--schedule` 指定的日期和時間 (2021 年 3 月 31 日下午 10 點 UTC)，如果 `MinCapacity` 指定的值高於目前的容量，則 Application Auto Scaling 會水平擴展至 `MinCapacity`。

Linux、macOS 或 Unix

```
aws application-autoscaling put-scheduled-action --service-namespace custom-resource \
  --scalable-dimension custom-resource:ResourceType:Property \
  --resource-id file://~/custom-resource-id.txt \
  --scheduled-action-name scale-out \
  --schedule "at(2021-03-31T22:00:00)" \
  --scalable-target-action MinCapacity=3
```

Windows

```
aws application-autoscaling put-scheduled-action --service-namespace custom-resource ^
  --scalable-dimension custom-resource:ResourceType:Property ^
  --resource-id file://~/custom-resource-id.txt ^
  --scheduled-action-name scale-out ^
  --schedule "at(2021-03-31T22:00:00)" ^
  --scalable-target-action MinCapacity=3
```

執行此排定的動作時，如果容量上限小於容量下限指定的值，則您必須指定新的容量上限和下限，而不只是容量下限。

Example 範例：僅縮減一次

以下是建立排定動作在特定日期和時間縮減容量的範例。

在 `--schedule` 指定的日期和時間 (2021 年 3 月 31 日下午 10 點 30 分 UTC)，如果 `MaxCapacity` 指定的值低於目前的容量，則 Application Auto Scaling 會縮減至 `MaxCapacity`。

Linux、macOS 或 Unix

```
aws application-autoscaling put-scheduled-action --service-namespace custom-resource \
  --scalable-dimension custom-resource:ResourceType:Property \
  --resource-id file://~/custom-resource-id.txt \
  --scheduled-action-name scale-in \
  --schedule "at(2021-03-31T22:30:00)" \
  --scalable-target-action MinCapacity=0,MaxCapacity=0
```

Windows

```
aws application-autoscaling put-scheduled-action --service-namespace custom-resource ^
  --scalable-dimension custom-resource:ResourceType:Property ^
  --resource-id file://~/custom-resource-id.txt ^
  --scheduled-action-name scale-in ^
  --schedule "at(2021-03-31T22:30:00)" ^
  --scalable-target-action MinCapacity=0,MaxCapacity=0
```

建立依週期性間隔執行的排定動作

若要依據週期性間隔來執行排程擴展，請使用 `--schedule "rate(value unit)"` 選項。其值必須為正整數。單位可以是 `minute`、`minutes`、`hour`、`hours`、`day` 或 `days`。如需詳細資訊，請參閱《Amazon EventBridge 使用者指南》中的 Rate 運算式。

以下是使用 Rate 表達式的排定動作範例。

按照指定的排程 (從 2021 年 1 月 30 日下午中午 12 點 UTC 開始至 2021 年 1 月 31 日下午 10 點 UTC 結束，每隔 5 小時)，如果 MinCapacity 指定的值高於目前的容量，則 Application Auto Scaling 會水平擴展至 MinCapacity。如果 MaxCapacity 指定的值低於目前的容量，則 Application Auto Scaling 會縮減至 MaxCapacity。

Linux、macOS 或 Unix

```
aws application-autoscaling put-scheduled-action --service-namespace ecs \
  --scalable-dimension ecs:service:DesiredCount \
  --resource-id service/my-cluster/my-service \
  --scheduled-action-name my-recurring-action \
  --schedule "rate(5 hours)" \
  --start-time 2021-01-30T12:00:00 \
  --end-time 2021-01-31T22:00:00 \
  --scalable-target-action MinCapacity=3,MaxCapacity=10
```

Windows

```
aws application-autoscaling put-scheduled-action --service-namespace ecs ^
  --scalable-dimension ecs:service:DesiredCount ^
  --resource-id service/my-cluster/my-service ^
  --scheduled-action-name my-recurring-action ^
  --schedule "rate(5 hours)" ^
  --start-time 2021-01-30T12:00:00 ^
  --end-time 2021-01-31T22:00:00 ^
  --scalable-target-action MinCapacity=3,MaxCapacity=10
```

建立依週期性排程執行的排程動作

若要依據週期性排程來執行排程擴展，請使用 --schedule "cron(*fields*)" 選項。如需詳細資訊，請參閱使用 Application Auto Scaling 排程週期性擴展動作。

以下是使用 Rate 表達式的排定動作範例。

按照指定的排程 (每天上午 9 點 UTC)，如果 MinCapacity 指定的值高於目前的容量，則 Application Auto Scaling 會水平擴展至 MinCapacity。如果 MaxCapacity 指定的值低於目前的容量，則 Application Auto Scaling 會縮減至 MaxCapacity。

Linux、macOS 或 Unix

```
aws application-autoscaling put-scheduled-action --service-namespace appstream \
  --scalable-dimension appstream:fleet:DesiredCapacity \
  --resource-id fleet/sample-fleet \
  --scheduled-action-name my-recurring-action \
  --schedule "cron(0 9 * * ? *)" \
  --scalable-target-action MinCapacity=10,MaxCapacity=50
```

Windows

```
aws application-autoscaling put-scheduled-action --service-namespace appstream ^
  --scalable-dimension appstream:fleet:DesiredCapacity ^
  --resource-id fleet/sample-fleet ^
  --scheduled-action-name my-recurring-action ^
  --schedule "cron(0 9 * * ? *)" ^
  --scalable-target-action MinCapacity=10,MaxCapacity=50
```

建立一次性排定動作並指定時區

排定的動作預設為 UTC 時區。若要指定不同的時區，請包含 `--timezone` 選項，並指定時區的正式名稱 (例如 `America/New_York`)。如需詳細資訊，請參閱 https://www.joda.org/joda-time/timezones.html，其中提供呼叫 put-scheduled-action 時所支援 IANA 時區的相關資訊。

以下是建立排定動作在特定日期和時間擴展容量時使用 `--timezone` 選項的範例。

在 `--schedule` 指定的日期和時間 (2021 年 1 月 31 日下午 5 點當地時間)，如果 `MinCapacity` 指定的值高於目前的容量，則 Application Auto Scaling 會水平擴展至 `MinCapacity`。如果 `MaxCapacity` 指定的值低於目前的容量，則 Application Auto Scaling 會縮減至 `MaxCapacity`。

Linux、macOS 或 Unix

```
aws application-autoscaling put-scheduled-action --service-namespace comprehend \
  --scalable-dimension comprehend:document-classifier-endpoint:DesiredInferenceUnits \
  --resource-id arn:aws:comprehend:us-west-2:123456789012:document-classifier-endpoint/
EXAMPLE \
  --scheduled-action-name  my-one-time-action \
  --schedule "at(2021-01-31T17:00:00)" --timezone "America/New_York" \
  --scalable-target-action MinCapacity=1,MaxCapacity=3
```

Windows

```
aws application-autoscaling put-scheduled-action --service-namespace comprehend ^
```

```
  --scalable-dimension comprehend:document-classifier-endpoint:DesiredInferenceUnits ^
  --resource-id arn:aws:comprehend:us-west-2:123456789012:document-classifier-endpoint/
EXAMPLE ^
  --scheduled-action-name  my-one-time-action ^
  --schedule "at(2021-01-31T17:00:00)" --timezone "America/New_York" ^
  --scalable-target-action MinCapacity=1,MaxCapacity=3
```

建立指定時區的週期性排程動作

以下範例使用 `--timezone` 選項，在建立週期性排程動作時擴展容量。如需詳細資訊，請參閱使用 Application Auto Scaling 排程週期性擴展動作。

按照指定的排程 (每週一到週五下午 6 點當地時間)，如果 `MinCapacity` 指定的值高於目前的容量，則 Application Auto Scaling 會水平擴展至 `MinCapacity`。如果 `MaxCapacity` 指定的值低於目前的容量，則 Application Auto Scaling 會縮減至 `MaxCapacity`。

Linux、macOS 或 Unix

```
aws application-autoscaling put-scheduled-action --service-namespace lambda \
  --scalable-dimension lambda:function:ProvisionedConcurrency \
  --resource-id function:my-function:BLUE \
  --scheduled-action-name my-recurring-action \
  --schedule "cron(0 18 ? * MON-FRI *)" --timezone "Etc/GMT+9" \
  --scalable-target-action MinCapacity=10,MaxCapacity=50
```

Windows

```
aws application-autoscaling put-scheduled-action --service-namespace lambda ^
  --scalable-dimension lambda:function:ProvisionedConcurrency ^
  --resource-id function:my-function:BLUE ^
  --scheduled-action-name my-recurring-action ^
  --schedule "cron(0 18 ? * MON-FRI *)" --timezone "Etc/GMT+9" ^
  --scalable-target-action MinCapacity=10,MaxCapacity=50
```

使用 描述 Application Auto Scaling 的排程擴展 AWS CLI

這些範例 AWS CLI 命令使用與 Application Auto Scaling 整合之 服務的資源，描述擴展活動和排程動作。對於不同的可擴展性目標，請在 中指定其命名空間`--service-namespace`、在 中指定其可擴展性維度`--scalable-dimension`，以及在 中指定其資源 ID`--resource-id`。

使用 時 AWS CLI，請記住，您的命令會在為設定檔 AWS 區域 設定的 中執行。如果您想在不同區域中執行命令，則可變更設定檔的預設區域，或搭配 `--region` 參數使用命令。

範例

- 描述服務的擴展活動
- 描述服務的排程動作
- 描述可擴展目標的排程動作

描述服務的擴展活動

若要檢視指定的服務命名空間中所有可擴展目標的擴展活動，請使用 describe-scaling-activities 命令。

以下範例會擷取與 dynamodb 服務命名空間相關聯的擴展活動。

Linux、macOS 或 Unix

```
aws application-autoscaling describe-scaling-activities --service-namespace dynamodb
```

Windows

```
aws application-autoscaling describe-scaling-activities --service-namespace dynamodb
```

輸出

如果命令成功，則會傳回類似以下的輸出。

```
{
    "ScalingActivities": [
        {
            "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
            "Description": "Setting write capacity units to 10.",
            "ResourceId": "table/my-table",
            "ActivityId": "4d1308c0-bbcf-4514-a673-b0220ae38547",
            "StartTime": 1561574415.086,
            "ServiceNamespace": "dynamodb",
            "EndTime": 1561574449.51,
            "Cause": "maximum capacity was set to 10",
            "StatusMessage": "Successfully set write capacity units to 10. Change
 successfully fulfilled by dynamodb.",
```

```
            "StatusCode": "Successful"
        },
        {
            "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
            "Description": "Setting min capacity to 5 and max capacity to 10",
            "ResourceId": "table/my-table",
            "ActivityId": "f2b7847b-721d-4e01-8ef0-0c8d3bacc1c7",
            "StartTime": 1561574414.644,
            "ServiceNamespace": "dynamodb",
            "Cause": "scheduled action name my-second-scheduled-action was triggered",
            "StatusMessage": "Successfully set min capacity to 5 and max capacity to
 10",
            "StatusCode": "Successful"
        },
        {
            "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
            "Description": "Setting write capacity units to 15.",
            "ResourceId": "table/my-table",
            "ActivityId": "d8ea4de6-9eaa-499f-b466-2cc5e681ba8b",
            "StartTime": 1561574108.904,
            "ServiceNamespace": "dynamodb",
            "EndTime": 1561574140.255,
            "Cause": "minimum capacity was set to 15",
            "StatusMessage": "Successfully set write capacity units to 15. Change
 successfully fulfilled by dynamodb.",
            "StatusCode": "Successful"
        },
        {
            "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
            "Description": "Setting min capacity to 15 and max capacity to 20",
            "ResourceId": "table/my-table",
            "ActivityId": "3250fd06-6940-4e8e-bb1f-d494db7554d2",
            "StartTime": 1561574108.512,
            "ServiceNamespace": "dynamodb",
            "Cause": "scheduled action name my-first-scheduled-action was triggered",
            "StatusMessage": "Successfully set min capacity to 15 and max capacity to
 20",
            "StatusCode": "Successful"
        }
    ]
}
```

若要將此命令變更為只擷取其中一個可擴展目標的擴展活動，請增加 `--resource-id` 選項。

描述服務的排程動作

若要描述指定的服務命名空間中所有可擴展目標的排定動作，請使用 describe-scheduled-actions 命令。

以下範例會擷取與 ec2 服務命名空間相關聯的排定動作。

Linux、macOS 或 Unix

```
aws application-autoscaling describe-scheduled-actions --service-namespace ec2
```

Windows

```
aws application-autoscaling describe-scheduled-actions --service-namespace ec2
```

輸出

如果命令成功，則會傳回類似以下的輸出。

```
{
    "ScheduledActions": [
        {
            "ScheduledActionName": "my-one-time-action",
            "ScheduledActionARN": "arn:aws:autoscaling:us-
west-2:123456789012:scheduledAction:493a6261-fbb9-432d-855d-3c302c14bdb9:resource/ec2/
spot-fleet-request/sfr-107dc873-0802-4402-a901-37294EXAMPLE:scheduledActionName/my-one-
time-action",
            "ServiceNamespace": "ec2",
            "Schedule": "at(2021-01-31T17:00:00)",
            "Timezone": "America/New_York",
            "ResourceId": "spot-fleet-request/sfr-107dc873-0802-4402-
a901-37294EXAMPLE",
            "ScalableDimension": "ec2:spot-fleet-request:TargetCapacity",
            "ScalableTargetAction": {
                "MaxCapacity": 1
            },
            "CreationTime": 1607454792.331
        },
        {
            "ScheduledActionName": "my-recurring-action",
            "ScheduledActionARN": "arn:aws:autoscaling:us-
west-2:123456789012:scheduledAction:493a6261-fbb9-432d-855d-3c302c14bdb9:resource/ec2/
```

```
spot-fleet-request/sfr-107dc873-0802-4402-a901-37294EXAMPLE:scheduledActionName/my-
recurring-action",
            "ServiceNamespace": "ec2",
            "Schedule": "rate(5 minutes)",
            "ResourceId": "spot-fleet-request/sfr-107dc873-0802-4402-
a901-37294EXAMPLE",
            "ScalableDimension": "ec2:spot-fleet-request:TargetCapacity",
            "StartTime": 1604059200.0,
            "EndTime": 1612130400.0,
            "ScalableTargetAction": {
                "MinCapacity": 3,
                "MaxCapacity": 10
            },
            "CreationTime": 1607454949.719
        },
        {
            "ScheduledActionName": "my-one-time-action",
            "ScheduledActionARN": "arn:aws:autoscaling:us-
west-2:123456789012:scheduledAction:4bce34c7-bb81-4ecf-b776-5c726efb1567:resource/ec2/
spot-fleet-request/sfr-40edeb7b-9ae7-44be-bef2-5c4c8EXAMPLE:scheduledActionName/my-one-
time-action",
            "ServiceNamespace": "ec2",
            "Schedule": "at(2020-12-08T9:36:00)",
            "Timezone": "America/New_York",
            "ResourceId": "spot-fleet-request/sfr-40edeb7b-9ae7-44be-
bef2-5c4c8EXAMPLE",
            "ScalableDimension": "ec2:spot-fleet-request:TargetCapacity",
            "ScalableTargetAction": {
                "MinCapacity": 1,
                "MaxCapacity": 3
            },
            "CreationTime": 1607456031.391
        }
    ]
}
```

描述可擴展目標的排程動作

對於特定可擴展目標的排定動作，若要擷取相關資訊，請在使用 describe-scheduled-actions 命令來描述排定的動作時，增加 `--resource-id` 選項。

如果您包含 `--scheduled-action-names` 選項並指定排定動作的名稱作為值，則此命令只會傳回名稱相符的排定動作，如下列範例所示。

Linux、macOS 或 Unix

```
aws application-autoscaling describe-scheduled-actions --service-namespace ec2 \
  --resource-id spot-fleet-request/sfr-40edeb7b-9ae7-44be-bef2-5c4c8EXAMPLE \
  --scheduled-action-names my-one-time-action
```

Windows

```
aws application-autoscaling describe-scheduled-actions --service-namespace ec2 ^
  --resource-id spot-fleet-request/sfr-40edeb7b-9ae7-44be-bef2-5c4c8EXAMPLE ^
  --scheduled-action-names my-one-time-action
```

輸出

如果命令成功，則會傳回類似以下的輸出。如果您為 提供多個值--scheduled-action-names，則輸出會包含名稱相符的所有排程動作。

```
{
    "ScheduledActions": [
        {
            "ScheduledActionName": "my-one-time-action",
            "ScheduledActionARN": "arn:aws:autoscaling:us-
west-2:123456789012:scheduledAction:4bce34c7-bb81-4ecf-b776-5c726efb1567:resource/ec2/
spot-fleet-request/sfr-40edeb7b-9ae7-44be-bef2-5c4c8EXAMPLE:scheduledActionName/my-one-
time-action",
            "ServiceNamespace": "ec2",
            "Schedule": "at(2020-12-08T9:36:00)",
            "Timezone": "America/New_York",
            "ResourceId": "spot-fleet-request/sfr-40edeb7b-9ae7-44be-
bef2-5c4c8EXAMPLE",
            "ScalableDimension": "ec2:spot-fleet-request:TargetCapacity",
            "ScalableTargetAction": {
                "MinCapacity": 1,
                "MaxCapacity": 3
            },
            "CreationTime": 1607456031.391
        }
    ]
}
```

使用 Application Auto Scaling 排程週期性擴展動作

> **Important**
>
> 如需適用於 Amazon EC2 Auto Scaling 之 cron 運算式的輔助說明，請參閱 《Amazon EC2 Auto Scaling 使用者指南》中的週期性排程主題。藉由 Amazon EC2 Auto Scaling，您就可以使用傳統的 cron 語法，而不是 Application Auto Scaling 使用的自訂 cron 語法。

您可以建立排定的動作，使用 cron 運算式依週期性排程執行。

若要建立週期性排程，請指定 cron 運算式和時區來描述該排定動作何時會重複發生。支援的時區值是 Joda-Time 支援的 IANA 時區標準名稱 (例如 `Etc/GMT+9` 或 `Pacific/Tahiti`)。您可以選擇性地為開始時間、結束時間 (或兩者) 指定日期和時間。如需使用 AWS CLI 建立排程動作的範例命令，請參閱建立指定時區的週期性排程動作。

受支援的 cron 運算式格式由六個以空格分隔的欄位組成：[分鐘] [小時] [一個月的第幾日] [月] [一週的第幾日] [年]。例如，Cron 表達式 `30 6 ? * MON *` 會設定排程動作，每週一上午 6:30 重複執行。使用星號作為萬用字元，以比對欄位的所有數值。

如需 Application Auto Scaling 排程動作之 Cron 語法的詳細資訊，請參閱《Amazon EventBridge 使用者指南》中的 Cron 表達式參考。

建立週期性排程時，請謹慎選擇開始時間與結束時間。請謹記以下幾點：

- 如果您指定開始時間，則 Application Auto Scaling 會在此時間執行動作，之後就根據指定的週期執行該動作。
- 如果指定了結束時間，過了此時刻會停止此動作。Application Auto Scaling 不會追蹤先前的值，在結束時間之後也不會回復到先前的那些值。
- 當您使用 AWS CLI 或 SDKs 建立或更新排程動作時， AWS 開始時間和結束時間必須以 UTC 設定。

範例

建立 Application Auto Scaling 可擴展目標的週期性排程時，您可以參考下表。以下範例是使用 Application Auto Scaling 建立或更新排定動作的正確語法。

分鐘	小時	月中的日	月	週中的日	年	意義
0	10	*	*	?	*	在每天上午 10:00 (UTC) 執行
15	12	*	*	?	*	在每天下午 12:15 (UTC) 執行
0	18	?	*	MON-FRI	*	在每週一至週五下午 6:00 (UTC) 執行
0	8	1	*	?	*	在每個月第 1 天上午 8 點 (UTC) 執行
0/15	*	*	*	?	*	每 15 分鐘執行
0/10	*	?	*	MON-FRI	*	在週一至週五每 10 分鐘執行
0/5	8-17	?	*	MON-FRI	*	在週一至週五上午 8:00 至下午 5:55 (UTC) 之間每 5 分鐘執行

異常情形

您還可以使用包含七個欄位的字串值建立 cron 運算式。在這種情況下，您可以使用前三個欄位來指定應執行排定動作的時間，包含秒。完整的 cron 運算式具有以下以空格分隔的欄位：[秒] [分鐘] [小時] [一個月的第幾日] [月] [一週的第幾日] [年]。但是，此方法並不能保證排定的動作會於您指定的精確秒數執行。此外，某些服務主控台可能不支援 cron 運算式中的秒欄位。

對可擴展的目標停用排定擴展

您可以暫時停用排程擴展而不刪除排定的動作。如需詳細資訊，請參閱Application Auto Scaling 暫停和繼續擴展。

暫停排程擴展

若要在可擴展的目標上暫停排程擴展，請使用 register-scalable-target 命令與 `--suspended-state` 選項，並將 `ScheduledScalingSuspended` 屬性的值指定為 `true`，如下列範例所示。

Linux、macOS 或 Unix

```
aws application-autoscaling register-scalable-target --service-namespace rds \
  --scalable-dimension rds:cluster:ReadReplicaCount --resource-id cluster:my-db-cluster
 \
  --suspended-state '{"ScheduledScalingSuspended": true}'
```

Windows

```
aws application-autoscaling register-scalable-target --service-namespace rds ^
  --scalable-dimension rds:cluster:ReadReplicaCount --resource-id cluster:my-db-cluster
 ^
  --suspended-state "{\"ScheduledScalingSuspended\": true}"
```

輸出

如果命令成功，則會傳回可擴展目標的 ARN。下列為範例輸出。

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-
target/1234abcd56ab78cd901ef1234567890ab123"
}
```

恢復排程擴展

若要繼續排程擴展，請再次執行 register-scalable-target命令，指定 `false`做為 的值`ScheduledScalingSuspended`。

使用 刪除 Application Auto Scaling 的排程動作 AWS CLI

排定動作完成後，您可將其刪除。

刪除您的排程動作

使用 delete-scheduled-action 命令。如果成功，此命令不會傳回任何輸出。

Linux、macOS 或 Unix

```
aws application-autoscaling delete-scheduled-action \
  --service-namespace ec2 \
  --scalable-dimension ec2:spot-fleet-request:TargetCapacity \
  --resource-id spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-37294EXAMPLE \
  --scheduled-action-name my-recurring-action
```

Windows

```
aws application-autoscaling delete-scheduled-action ^
  --service-namespace ec2 ^
  --scalable-dimension ec2:spot-fleet-request:TargetCapacity ^
  --resource-id spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-37294EXAMPLE ^
  --scheduled-action-name my-recurring-action
```

解除登錄可擴展的目標

如果您也已完成可擴展的目標，則可以取消註冊。使用下列 deregister-scalable-target 命令。如果有任何尚未刪除的擴展政策或排程動作，則會由此命令刪除。如果成功，此命令不會傳回任何輸出。

Linux、macOS 或 Unix

```
aws application-autoscaling deregister-scalable-target \
  --service-namespace ec2 \
  --scalable-dimension ec2:spot-fleet-request:TargetCapacity \
  --resource-id spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-37294EXAMPLE
```

Windows

```
aws application-autoscaling deregister-scalable-target ^
  --service-namespace ec2 ^
  --scalable-dimension ec2:spot-fleet-request:TargetCapacity ^
  --resource-id spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-37294EXAMPLE
```

Application Auto Scaling 的目標追蹤擴展政策

目標追蹤擴展政策會根據目標指標的值，自動擴展應用程式規模。如此一來，您的應用程式就能維持最佳效能和成本效益，無需手動介入。

如果使用目標追蹤，您必須選取指標和目標值，以代表應用程式理想的平均使用率或輸送量。Application Auto Scaling 會建立和管理 CloudWatch 警示，當指標偏離目標就會觸發擴展事件，類似於電熱器會維持於目標溫度一樣。

例如，假設您目前有一個應用程式在 Spot 機群上執行，而且您希望當應用程式的負載變更時，該機群的 CPU 使用率保持在 50% 左右。這樣可以給予您額外的容量處理流量尖峰，而不會維持過多的閒置資源數。

您可以透過建立以 50% 的平均 CPU 利用率為目標的目標追蹤擴展政策來滿足此需求。當 CPU 為了處理增加的負載而使用率超過 50%，Application Auto Scaling 就會擴增規模 (增加容量)。當 CPU 使用率降至 50% 以下，則會縮減規模 (減少容量)，在低使用率期間維持最佳化成本。

目標追蹤政策無需手動定義 CloudWatch 警示和擴展調整內容，Application Auto Scaling 會根據您設定的目標自動處理。

您可以在預先定義或自訂指標的基礎上建立目標追蹤政策。

- 預先定義指標：Application Auto Scaling 提供的指標，例如平均 CPU 使用率或單一目標平均請求數。
- 自訂指標：您可以使用指標數學來合併指標、運用現有指標，或使用您自行發佈到 CloudWatch 的自訂指標。

選擇與可擴充目標容量變動情形依比例反向變化的指標。因此，如果您加倍容量，指標會減少 50%。如此一來，指標資料就能依比例準確觸發擴展事件。

目錄

Application Auto Scaling 的目標追蹤擴展如何運作

本主題說明目標追蹤擴展的運作方式，並介紹目標追蹤擴展政策的關鍵元素。

目錄

運作方式

若要使用目標追蹤擴展，您可以建立目標追蹤擴展政策，並指定下列項目：

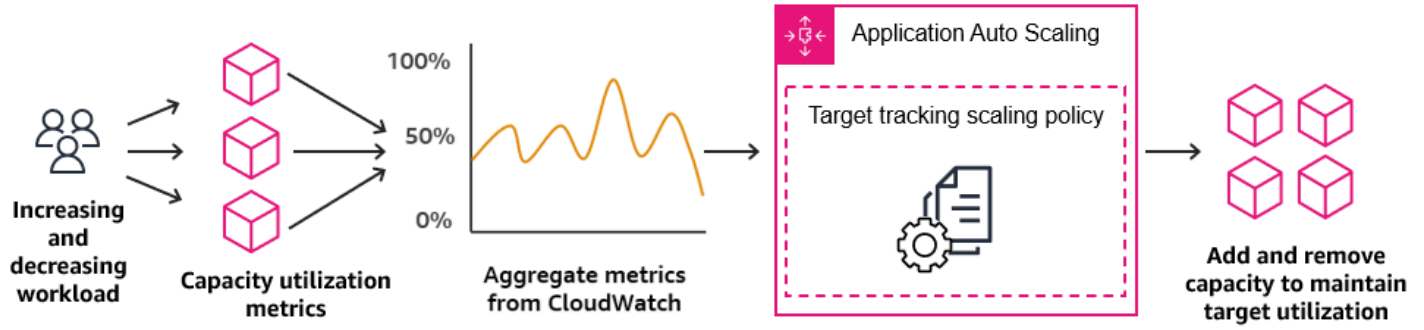
- 指標：要追蹤的 CloudWatch 指標，例如平均 CPU 使用率或單一目標平均請求數。
- 目標值：指標的目標值，例如 50% CPU 使用率或每分鐘每個目標 1000 個請求。

Application Auto Scaling 會建立和管理可觸發擴展政策的 CloudWatch 警示，並根據指標和目標值來計算擴展調整內容。該功能會視需求新增及移除容量，讓指標保持在等於或接近指定目標值的狀態。

指標高於目標值時，Application Auto Scaling 會擴增規模，亦即增加容量，以減少指標值與目標值之間的差距；當指標低於目標值時，Application Auto Scaling 則會移除容量，縮減規模。

擴展活動之間會有冷卻時間，以防止容量快速波動。您可自行選擇是否設定擴展政策的冷卻時間。

下圖顯示設置完成後目標追蹤擴展政策的運作方式總覽。



請注意，相較於在使用率減少時移除容量，目標追蹤擴展政策會在使用率增加時更積極地新增容量。例如，如果政策的指定指標達到目標值，政策會假設應用程式已經滿載。因此，它採取的回應方式為根據指標值的比例盡快新增容量。指標越高，新增的容量越多。

當指標低於目標值時，政策會預期使用率最終將再次增加。在此情況中，只有在使用率突破閾值遠低於目標值 (通常低於 10%)，而確定使用率減緩時，才會移除容量，擴展速度會變慢。這種較為保守的行為的目的是，確保只有在應用程式不再遇到與之前同樣大量的需求時，才會移除容量。

選擇 Metrics (指標)

您可以使用預先定義的指標或自訂指標建立目標追蹤擴展政策。

在使用預先定義的指標類型建立目標追蹤擴展政策時，您從 目標追蹤擴展政策的預先定義指標 中的預先定義指標清單選擇一個指標。

選擇指標時請謹記以下事項：

- 並非所有的自訂指標都適用於目標追蹤。指標必須是有效的使用率指標，而且能夠表示可擴展性目標的忙碌程度。指標值必須根據可擴展性目標的容量按比例增加或減少，以使指標資料可用於按比例擴展可擴展性目標。
- 若要使用 `ALBRequestCountPerTarget` 指標，您必須指定 `ResourceLabel` 參數來識別與指標相關聯的目標群組。
- 當指標向 CloudWatch 真的發出 0 值時 (例如 `ALBRequestCountPerTarget`)，如果應用程式持續一段時間沒有流量，Application Auto Scaling 可能縮減到 0。為了讓可擴展的目標在沒有收到請求時縮減到 0，可擴展目標的容量下限必須設定為 0。
- 您可以使用指標數學來合併現有的指標，而不是發布新的指標來用於您的擴展政策。如需詳細資訊，請參閱使用指標數學運算建立 Application Auto Scaling 的目標追蹤擴展政策。
- 若要查看使用的服務是否支援在服務主控台中指定自訂指標，請參閱該服務的文件。
- 建議您使用可以一分鐘間隔提供的指標，從而幫助您更快速地擴展以回應利用率變更。目標追蹤會針對所有預先定義的指標和自訂指標，評估以一分鐘精細度彙總的指標，但基礎指標可能會以較低頻

率發佈資料。例如，依預設，所有 Amazon EC2 指標都會以五分鐘的間隔傳送，但可設定為一分鐘 (稱為詳細監控)。此選擇取決於個別服務。大多數人會嘗試使用盡可能小的間隔。

定義目標值

建立目標追蹤擴展政策時，您必須指定目標值。目標值代表應用程式的最佳平均使用率或輸送量。若要以符合成本效益的方式使用資源，請盡可能高地設定目標值，並為非預期的流量增加提供合理的緩衝區。如果您的應用程式已經針對一般流量進行最佳化橫向擴展，實際指標值應等於或低於目標值。

擴展政策以輸送量為基礎時 (例如 Application Load Balancer 每個目標的要求計數、網路 I/O 或其他計數指標)，目標值代表一分鐘期間單一實體的最佳平均輸送量 (例如 Application Load Balancer 目標群組的單一目標)。

定義冷卻時間

您可以選擇在目標追蹤擴展政策中定義冷卻時間。

冷卻時間指定擴展政策等候上一個擴展活動生效的時間量。

冷卻時間有兩種類型：

- 向外擴展冷卻期間的目的是連續的規模擴展 (但並非過度)。在 Application Auto Scaling 使用擴展政策成功擴展之後，就會開始計算冷卻時間。除非觸發較大的橫向擴展或冷卻時間結束，否則擴展規模政策不會再次增加所需的容量。在向外擴展冷卻期間有效時，由起始冷卻的向外擴展活動所增加的容量，將計入下次向外擴展活動所需的容量。
- 縮減冷卻時間的用意在於保守縮減以保護應用程式的可用性，所以縮減冷卻期過後才會開放縮減活動。不過，若在向內擴展冷卻期間另有警示觸發了向外擴展活動，Application Auto Scaling 則會立即向外擴展目標。在這種情況下，縮減冷卻時間隨即停止，且不會完成。

每個冷卻期間都是以秒為單位進行測量，且僅適用於擴展政策相關擴展活動。在冷卻期間，當已排定的動作在已排定的時間開始時，它可以立即觸發擴展活動，而無須等候冷卻期間過期。

您可以從預設值開始，之後再進行微調。例如，您可能需要增加冷卻期間，以防止目標追蹤擴展政策對短時間內發生的變更過於積極。

預設值

Application Auto Scaling 為 ElastiCache 提供預設值 600，並為下列可擴展目標提供預設值 300：

- AppStream 2.0 機群
- Aurora 資料庫叢集
- ECS 服務
- Neptune 叢集
- SageMaker AI 端點變體
- SageMaker AI 推論元件
- SageMaker AI Serverless 佈建並行
- Spot Fleets
- WorkSpaces 集區
- 自訂資源

對於所有其他可擴展目標，預設值為 0 或 null：

- Amazon Comprehend 文件分類和實體識別器端點
- DynamoDB 資料表和全域次要索引
- Amazon Keyspaces 資料表
- Lambda 佈建並行
- Amazon MSK 代理程式儲存

當 Application Auto Scaling 評估冷卻時間時，Null 值會被視為與零值相同。

您可以更新任何預設值，包括 null 值，以設定自己的冷卻時間。

考量事項

使用目標追蹤擴展政策時，下列考量適用：

- 請勿編輯或刪除用於目標追蹤擴展政策的 CloudWatch 警示。Application Auto Scaling 可建立和管理與目標追蹤擴展政策關聯的 CloudWatch 警示，並會在不再需要時刪除它們。
- 如果指標缺少資料點，這會導致 CloudWatch 警示狀態變更為 `INSUFFICIENT_DATA`。發生這種情況時，Application Auto Scaling 無法擴展您的可擴展目標，直到找到新的資料點為止。如需詳細資訊，請參閱「Amazon CloudWatch 使用者指南」中的設定 CloudWatch 警示如何處理遺失資料。
- 如果指標在設計上是以稀疏的方式來報告，指標數學可能會有所幫助。例如，若要使用最新的值，請使用指標為 `m1` 的 `FILL(m1,REPEAT)` 函數。

- 您可能會看到目標值與實際指標資料點之間有些差距。原因是 Application Auto Scaling 在決定新增或移除多少容量時，一律以四捨五入來保守處理。這樣可防止新增不足的容量，或移除過多的容量。不過，對於具有小容量的可擴展性目標，實際指標資料點可能會遠於目標值。

  對於具有較大容量的可擴展性目標，新增或移除容量促使目標值與實際指標資料點之間的差距變少。
- 目標追蹤擴展政策假設在指定的指標超過目標值時，應執行向外擴展。您無法使用目標追蹤擴展政策在指定的指標低於目標值時執行向外擴展。

多個擴展政策

任一個可擴展性目標均能有多個目標追蹤擴展政策，但前提是各政策使用不同的指標。Application Auto Scaling 的用意一律以可用性為優先，因此其行為視目標追蹤政策是準備水平擴展或縮減而有所不同。如果任何目標追蹤政策已準備好擴展，此服務就會擴展可擴展目標，但只有在所有目標追蹤政策 (已啟用縮減部分) 已準備好縮減，才會進行縮減。

如果多個擴展政策同一時間指示可擴展的目標橫向擴展或縮減，Application Auto Scaling 會根據縮減和橫向擴展都可達最大容量的政策來擴展。如此能夠更靈活地涵蓋多種情況，確保始終有足夠的容量可處理您的工作負載。

您可以停用目標追蹤擴展政策的縮減部分，以對縮減和橫向擴展使用不同的方法。例如，您可以使用步進擴展政策進行向內擴展，同時使用目標追蹤擴展政策向外擴展。

不過，我們建議您小心使用目標追蹤擴展政策與步進擴展政策，因為這些政策之間的衝突可能會造成非預期行為。例如，如果步進擴展政策在目標追蹤政策準備好縮減之前即啟動縮減活動，則不會封鎖縮減活動。向內擴展活動完成之後，目標追蹤政策可以指示可擴展的目標再次向外擴展。

對於本質上是循環的工作負載，您也可以選擇使用已排定的擴展在排程上自動執行容量變更。對於每個已排定的動作，可以定義新的容量下限值和新的容量上限值。這些值形成擴展政策的界限。透過結合已排定的擴展和目標追蹤擴展，可協助在立即需要容量時，降低使用率急遽增加的影響。

建立、管理及刪除擴展政策常用的命令

擴展政策常用的命令包括：

- register-scalable-target 將 AWS 或自訂資源註冊為可擴展的目標 (Application Auto Scaling 可以擴展的資源），以及暫停和繼續擴展。
- put-calable-policy，新增或修改現有可擴展目標的擴展政策。
- describe-scaling-activities 傳回 AWS 區域中擴展活動的相關資訊。
- describe-scaling-policies，傳回 AWS 區域的擴展政策相關資訊。

- delete-scaling-policy，刪除擴展政策。

相關資源

如需取得有關建立 Amazon EC2 Auto Scaling 的目標追蹤擴展政策之資訊，請參閱《Amazon EC2 Auto Scaling 使用者指南》中的「Amazon EC2 Auto Scaling 的目標追蹤擴展政策」。

限制

以下是使用目標追蹤擴展政策時的限制：

- 可擴展的目標不能是 Amazon EMR 叢集。Amazon EMR 不支援目標追蹤擴展政策。
- 當 Amazon MSK 叢集是可擴展的目標時，縮減會停用且無法啟用。
- 您無法使用 `RegisterScalableTarget`或 `PutScalingPolicy` API 操作來更新 AWS Auto Scaling 擴展計劃。
- 主控台對可擴展資源的檢視、新增、更新或移除目標追蹤擴展政策的存取，視您所使用的資源而定。如需詳細資訊，請參閱AWS 服務 可與 Application Auto Scaling 搭配使用。

使用 建立 Application Auto Scaling 的目標追蹤擴展政策 AWS CLI

此範例使用 AWS CLI 命令來建立 Amazon EC2 Spot 機群的目標機架政策。對於不同的可擴展性目標，請在 中指定其命名空間`--service-namespace`、在 中指定其可擴展性維度`--scalable-dimension`，以及在 中指定其資源 ID`--resource-id`。

使用 時 AWS CLI，請記住，您的命令會在為設定檔 AWS 區域 設定的 中執行。如果您想在不同區域中執行命令，則可變更設定檔的預設區域，或搭配 `--region` 參數使用命令。

任務

- 步驟 1：註冊可擴展的目標
- 步驟 2：建立目標追蹤擴展政策
- 步驟 3：描述目標追蹤擴展政策

步驟 1：註冊可擴展的目標

如果您尚未註冊可擴展的目標，請註冊。使用 register-scalable-target 命令，將目標服務中的特定資源註冊為可擴展的目標。以下範例向 Application Auto Scaling 註冊 Spot 機群請求。Application Auto

Scaling 在 Spot 機群中可擴展的執行個體數量下限為 2 個執行個體，上限為 10 個執行個體。將每個*########*替換為自己的資訊。

Linux、macOS 或 Unix

```
aws application-autoscaling register-scalable-target --service-namespace ec2 \
  --scalable-dimension ec2:spot-fleet-request:TargetCapacity \
  --resource-id spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE \
  --min-capacity 2 --max-capacity 10
```

Windows

```
aws application-autoscaling register-scalable-target --service-namespace ec2 ^
  --scalable-dimension ec2:spot-fleet-request:TargetCapacity ^
  --resource-id spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE ^
  --min-capacity 2 --max-capacity 10
```

輸出

如果成功，此命令會傳回可擴展目標的 ARN。下列為範例輸出。

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-
target/1234abcd56ab78cd901ef1234567890ab123"
}
```

步驟 2：建立目標追蹤擴展政策

若要建立目標追蹤擴展政策，您可以使用下列範例來協助您開始使用。

建立目標追蹤擴展政策

1. 使用下列cat命令，將擴展政策的目標值和預先定義的指標規格存放在主目錄中名為config.json的 JSON 檔案中。以下是將平均 CPU 使用率保持在 50% 的目標追蹤組態範例。

```
$ cat ~/config.json
{
  "TargetValue": 50.0,
  "PredefinedMetricSpecification":
    {
      "PredefinedMetricType": "EC2SpotFleetRequestAverageCPUUtilization"
```

```
    }
}
```

如需詳細資訊，請參閱《Application Auto Scaling API 參考》中的 PredefinedMetricSpecification。

或者，您可以建立自訂指標規格並從 CloudWatch 新增每個參數的值，以使用自訂的擴展指標。以下是將指定指標的平均使用率保持在 100 的目標追蹤組態範例。

```
$ cat ~/config.json
{
   "TargetValue": 100.0,
   "CustomizedMetricSpecification":{
      "MetricName": "MyUtilizationMetric",
      "Namespace": "MyNamespace",
      "Dimensions": [
         {
            "Name": "MyOptionalMetricDimensionName",
            "Value": "MyOptionalMetricDimensionValue"
         }
      ],
      "Statistic": "Average",
      "Unit": "Percent"
   }
}
```

如需詳細資訊，請參閱《Application Auto Scaling API 參考》中的 CustomizedMetricSpecification。

2. 使用以下 put-scaling-policy 命令並指定您建立的 `config.json` 檔案，建立名為 `cpu50-target-tracking-scaling-policy` 的擴展政策。

   Linux、macOS 或 Unix

```
aws application-autoscaling put-scaling-policy --service-namespace ec2 \
  --scalable-dimension ec2:spot-fleet-request:TargetCapacity \
  --resource-id spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE \
  --policy-name cpu50-target-tracking-scaling-policy --policy-type
 TargetTrackingScaling \
  --target-tracking-scaling-policy-configuration file://config.json
```

   Windows

```
aws application-autoscaling put-scaling-policy --service-namespace ec2 ^
  --scalable-dimension ec2:spot-fleet-request:TargetCapacity ^
  --resource-id spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE ^
  --policy-name cpu50-target-tracking-scaling-policy --policy-type
 TargetTrackingScaling ^
  --target-tracking-scaling-policy-configuration file://config.json
```

輸出

如果成功，此命令會傳回代表您建立的兩個 CloudWatch 警示的 ARN 和名稱。下列為範例輸出。

```
{
    "PolicyARN": "arn:aws:autoscaling:region:account-
id:scalingPolicy:policy-id:resource/ec2/spot-fleet-request/sfr-73fbd2ce-
aa30-494c-8788-1cee4EXAMPLE:policyName/cpu50-target-tracking-scaling-policy",
    "Alarms": [
        {
            "AlarmARN": "arn:aws:cloudwatch:region:account-id:alarm:TargetTracking-
spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE-AlarmHigh-d4f0770c-
b46e-434a-a60f-3b36d653feca",
            "AlarmName": "TargetTracking-spot-fleet-request/sfr-73fbd2ce-
aa30-494c-8788-1cee4EXAMPLE-AlarmHigh-d4f0770c-b46e-434a-a60f-3b36d653feca"
        },
        {
            "AlarmARN": "arn:aws:cloudwatch:region:account-id:alarm:TargetTracking-
spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE-AlarmLow-1b437334-
d19b-4a63-a812-6c67aaf2910d",
            "AlarmName": "TargetTracking-spot-fleet-request/sfr-73fbd2ce-
aa30-494c-8788-1cee4EXAMPLE-AlarmLow-1b437334-d19b-4a63-a812-6c67aaf2910d"
        }
    ]
}
```

步驟 3：描述目標追蹤擴展政策

您可使用以下 describe-scaling-policies 命令，描述指定的服務命名空間的所有擴展政策。

```
aws application-autoscaling describe-scaling-policies --service-namespace ec2
```

使用 --query 參數可將結果篩選為僅包含目標追蹤擴展政策。如需 query 語法的詳細資訊，請參閱《AWS Command Line Interface 使用者指南》中的從 AWS CLI控制命令輸出。

Linux、macOS 或 Unix

```
aws application-autoscaling describe-scaling-policies --service-namespace ec2 \
  --query 'ScalingPolicies[?PolicyType==`TargetTrackingScaling`]'
```

Windows

```
aws application-autoscaling describe-scaling-policies --service-namespace ec2 ^
  --query "ScalingPolicies[?PolicyType==`TargetTrackingScaling`]"
```

輸出

下列為範例輸出。

```
[
    {
        "PolicyARN": "PolicyARN",
        "TargetTrackingScalingPolicyConfiguration": {
            "PredefinedMetricSpecification": {
                "PredefinedMetricType": "EC2SpotFleetRequestAverageCPUUtilization"
            },
            "TargetValue": 50.0
        },
        "PolicyName": "cpu50-target-tracking-scaling-policy",
        "ScalableDimension": "ec2:spot-fleet-request:TargetCapacity",
        "ServiceNamespace": "ec2",
        "PolicyType": "TargetTrackingScaling",
        "ResourceId": "spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE",
        "Alarms": [
            {
                "AlarmARN": "arn:aws:cloudwatch:region:account-id:alarm:TargetTracking-
spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE-AlarmHigh-d4f0770c-
b46e-434a-a60f-3b36d653feca",
                "AlarmName": "TargetTracking-spot-fleet-request/sfr-73fbd2ce-
aa30-494c-8788-1cee4EXAMPLE-AlarmHigh-d4f0770c-b46e-434a-a60f-3b36d653feca"
            },
            {
                "AlarmARN": "arn:aws:cloudwatch:region:account-id:alarm:TargetTracking-
spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE-AlarmLow-1b437334-
d19b-4a63-a812-6c67aaf2910d",
```

```
                "AlarmName": "TargetTracking-spot-fleet-request/sfr-73fbd2ce-
aa30-494c-8788-1cee4EXAMPLE-AlarmLow-1b437334-d19b-4a63-a812-6c67aaf2910d"
            }
        ],
        "CreationTime": 1515021724.807
    }
]
```

使用 刪除 Application Auto Scaling 的目標追蹤擴展政策 AWS CLI

目標追蹤擴展政策用畢之後，您可以使用 delete-scaling-policy 命令將其刪除。

以下命令將刪除對指定的 Spot 機群請求所指定的目標追蹤擴展政策。還會刪除 Application Auto Scaling 替您建立的 CloudWatch 警示。

Linux、macOS 或 Unix

```
aws application-autoscaling delete-scaling-policy --service-namespace ec2 \
  --scalable-dimension ec2:spot-fleet-request:TargetCapacity \
  --resource-id spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE \
  --policy-name cpu50-target-tracking-scaling-policy
```

Windows

```
aws application-autoscaling delete-scaling-policy --service-namespace ec2 ^
  --scalable-dimension ec2:spot-fleet-request:TargetCapacity ^
  --resource-id spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE ^
  --policy-name cpu50-target-tracking-scaling-policy
```

使用指標數學運算建立 Application Auto Scaling 的目標追蹤擴展政策

利用指標數學，可查詢多個 CloudWatch 指標，並使用數學表達式根據這些指標來建立新的時間序列。您可以在 CloudWatch 主控台視覺化產生的時間序列，並將其新增至儀表板。如需有關指標數學的詳細資訊，請參閱《Amazon CloudWatch 使用者指南》中的使用指標數學。

指標數學運算式有下列考量：

- 您可以查詢任何可用的 CloudWatch 指標。每個指標都是指標名稱、命名空間以及零個或多個維度的唯一組合。

- 您可以使用任何算術運算子 (+-*/^)、統計函數 (如 AVG 或 SUM) 或 CloudWatch 支援的其他函數。
- 您可以在數學運算式的公式中同時使用其他數學運算式的指標和結果。
- 在指標規範中使用的任何運算式都必須最終傳回單一的時間序列。
- 您可以透過使用 CloudWatch 主控台或 CloudWatch GetMetricData API 驗證指標數學表達式是否有效。

主題

範例：每個任務的 Amazon SQS 佇列待辦項目

若要計算每個任務的 Amazon SQS 佇列待辦項目，請獲取佇列中可擷取的大致訊息數量，然後將該數字除以服務中執行的 Amazon ECS 任務數。如需詳細資訊，請參閱 AWS 運算部落格上的使用自訂指標的 Amazon Elastic Container Service (ECS) Auto Scaling。

運算式的邏輯如下所示：

```
sum of (number of messages in the queue)/(number of tasks that are
currently in the RUNNING state)
```

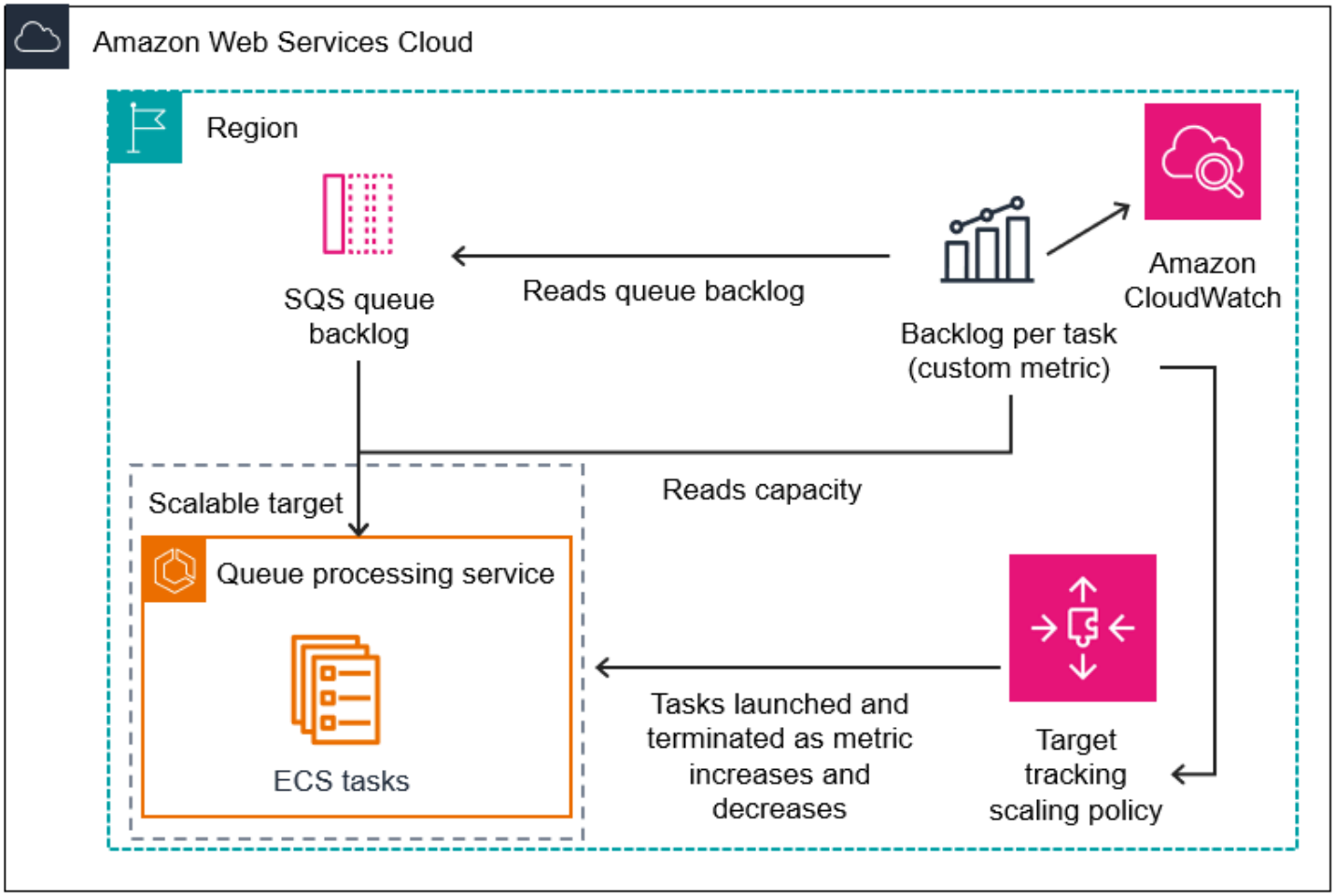
那麼，您的 CloudWatch 指標資訊如下。

ID	CloudWatch 指標	統計數字	期間
m1	ApproximateNumberOfMessagesVisible	總和	1 分鐘
m2	RunningTaskCount	平均數	1 分鐘

您的指標數學 ID 和表達式如下。

ID	表達式
e1	`(m1)/(m2)`

下圖說明此指標的架構：



使用此指標數學建立目標追蹤擴展政策 (AWS CLI)

1. 將指標數學運算式作為自訂指標規格的一部分儲存在名為 `config.json` 的 JSON 檔案中。

   使用以下範例協助您開始使用。將每個`#########`替換為自己的資訊。

```
{
    "CustomizedMetricSpecification": {
        "Metrics": [
            {
                "Label": "Get the queue size (the number of messages waiting to be
 processed)",
                "Id": "m1",
                "MetricStat": {
                    "Metric": {
                        "MetricName": "ApproximateNumberOfMessagesVisible",
                        "Namespace": "AWS/SQS",
```

```
                        "Dimensions": [
                            {
                                "Name": "QueueName",
                                "Value": "my-queue"
                            }
                        ]
                    },
                    "Stat": "Sum"
                },
                "ReturnData": false
            },
            {
                "Label": "Get the ECS running task count (the number of currently
running tasks)",
                "Id": "m2",
                "MetricStat": {
                    "Metric": {
                        "MetricName": "RunningTaskCount",
                        "Namespace": "ECS/ContainerInsights",
                        "Dimensions": [
                            {
                                "Name": "ClusterName",
                                "Value": "my-cluster"
                            },
                            {
                                "Name": "ServiceName",
                                "Value": "my-service"
                            }
                        ]
                    },
                    "Stat": "Average"
                },
                "ReturnData": false
            },
            {
                "Label": "Calculate the backlog per instance",
                "Id": "e1",
                "Expression": "m1 / m2",
                "ReturnData": true
            }
        ]
    },
    "TargetValue": 100
```

```
}
```

如需詳細資訊，請參閱 Application Auto Scaling API 參考中的 TargetTrackingScalingPolicyConfiguration。

> **Note**
>
> 以下是一些其他資源，可協助您尋找 CloudWatch 指標的指標名稱、命名空間、維度和統計資料：
>
> - 如需 AWS 服務可用指標的相關資訊，請參閱《Amazon AWS CloudWatch 使用者指南》中的發佈 CloudWatch 指標的 服務。 Amazon CloudWatch
> - 若要使用 取得 CloudWatch 指標的確切指標名稱、命名空間和維度 （ 如適用 ） AWS CLI，請參閱 清單指標。

2. 若要建立此政策，執行使用 JSON 檔案作為輸入 put-scaling-policy 命令，如以下範例所示。

```
aws application-autoscaling put-scaling-policy --policy-name sqs-backlog-target-
tracking-scaling-policy \
  --service-namespace ecs --scalable-dimension ecs:service:DesiredCount --resource-
id service/my-cluster/my-service \
  --policy-type TargetTrackingScaling --target-tracking-scaling-policy-
configuration file://config.json
```

如果成功，則此命令會傳回政策的 Amazon Resource Name (ARN) 和代表您建立的兩個 CloudWatch 警示的 ARN。

```
{
    "PolicyARN": "arn:aws:autoscaling:us-west-2:012345678910:scalingPolicy:
 8784a896-b2ba-47a1-b08c-27301cc499a1:resource/ecs/service/my-cluster/my-
service:policyName/sqs-backlog-target-tracking-scaling-policy",
    "Alarms": [
        {
            "AlarmARN": "arn:aws:cloudwatch:us-
west-2:012345678910:alarm:TargetTracking-service/my-cluster/my-service-
AlarmHigh-9bc77b56-0571-4276-ba0f-d4178882e0a0",
            "AlarmName": "TargetTracking-service/my-cluster/my-service-
AlarmHigh-9bc77b56-0571-4276-ba0f-d4178882e0a0"
        },
        {
```

```
            "AlarmARN": "arn:aws:cloudwatch:us-
west-2:012345678910:alarm:TargetTracking-service/my-cluster/my-service-
AlarmLow-9b6ad934-6d37-438e-9e05-02836ddcbdc4",
            "AlarmName": "TargetTracking-service/my-cluster/my-service-
AlarmLow-9b6ad934-6d37-438e-9e05-02836ddcbdc4"
        }
    ]
}
```

> **Note**
>
> 如果此命令擲回錯誤，請確定您已在 AWS CLI 本機將 更新為最新版本。

限制

- 要求大小上限為 50 KB。這是當您在政策定義中使用指標數學運算時 PutScalingPolicy API 要求的總承載大小。如果您超過此限制，Application Auto Scaling 會拒絕要求。
- 對目標追蹤擴展政策使用指標數學運算時，不支援下列服務：
  - Amazon Keyspaces (適用於 Apache Cassandra)
  - DynamoDB
  - Amazon EMR
  - Amazon MSK
  - Amazon Neptune

Application Auto Scaling 的步驟擴展政策

步進擴展政策會根據 CloudWatch 警示，依預先定義的增量幅度擴展應用程式的容量。您可以定義個別的擴展政策，在流量達到警示閾值時擴增規模 (增加容量) 和縮減規模 (減少容量)。

如果使用步進擴展政策，您可以建立及管理觸發擴展程序的 CloudWatch 警示。流量達到警示閾值時，Application Auto Scaling 就會啟動與該警示相關聯的擴展政策。

步進擴展政策會依據一組調整策略來調整容量，稱為步進調整。調整幅度會依流量達到警示閾值的程度而有所不同。

- 如果超過第一個閾值，Application Auto Scaling 會套用第一步調整。
- 如果超過第二個閾值，Application Auto Scaling 會套用第二步調整，以此類推。

如此一來，擴展政策就能適當回應警示指標的次要和重大變動。

此政策會繼續回應流量達到警示閾值的其他事件，即便是在擴展活動進行期間也不例外。換句話說，Application Auto Scaling 會在流量達到警示閾值時，評估所有警示事件。冷卻時間的作用在於防止流量快速而連續地多次達到警示閾值，導致過度擴展。

如同目標追蹤，步進擴展可協助您在流量有所變動時，自動擴展應用程式的容量。不過，如果擴展需求穩定，目標追蹤政策通常較容易實作及管理。

支援的可擴展目標

您可以將步進擴展政策與下列可擴展目標搭配使用：

- AppStream 2.0 機群
- Aurora 資料庫叢集
- ECS 服務
- EMR 叢集
- SageMaker AI 端點變體
- SageMaker AI 推論元件
- SageMaker AI Serverless 佈建並行
- Spot Fleets
- 自訂資源

目錄

Application Auto Scaling 的步驟擴展如何運作

本主題說明步驟擴展的運作方式，並介紹步驟擴展政策的關鍵元素。

目錄

運作方式

若要使用步進擴展，您必須建立 CloudWatch 警示，以監控可擴展的目標。定義指標、閾值和評估流量是否達到警示閾值的時段數。您也可以建立步進擴展政策，定義在流量達到警示閾值時如何擴展容量，並將其與可擴展目標建立關聯。

在政策中新增步進調整內容。您可以根據流量達到警示閾值的程度，定義不同的步進調整幅度。例如：

- 如果警示指標達到 60%，則擴增 10 個容量單位
- 如果警示指標達到 75%，則擴增 30 個容量單位
- 如果警示指標達到 85%，則擴增 40 個容量單位

流量在指定的評估時段達到警示閾值時，Application Auto Scaling 就會套用您在政策中定義的步進調整指示。警示狀態恢復 OK 後，才能依其他達標警示事件繼續調整。

擴展活動之間會有冷卻時間，以防止容量快速波動。您可自行選擇是否設定擴展政策的冷卻時間。

步驟調整

建立步進擴展政策時，您可以指定一或多個步進調整，這些調整會根據警示違規的程度自動動態調整目標的容量。每項步進調整可指定下列項目：

- 指標值下限
- 指標值上限
- 要擴展的數量，會以擴展調整類型為依據

CloudWatch 會根據 CloudWatch 警示相關聯指標的統計數字，彙總指標資料點。 超出警示閾值時，會呼叫適當的擴展政策。Application Auto Scaling 會將指定的彙總類型套用到 CloudWatch 最近的指標資料點 (而不是原始指標資料)。它會將彙總指標值與步進調整定義的上限和下限進行比較，以決定要執行哪一項步進調整。

您可以指定相對於違規閾值的上限及下限。例如，假設您設定流量超過 50% 指標時，由 CloudWatch 發出警示並執行擴增政策。當流量低於 50% 指標時，系統發出第二個警示，並執行縮減政策。每個政策都必須設定一組步進調整內容，調整類型為 `PercentChangeInCapacity`：

範例：擴增政策的步進調整

下限	上限	調整
0	10	0
10	20	10
20	無	30

範例：縮減政策的步進調整

下限	上限	調整
-10	0	0
-20	-10	-10
null	-20	-30

這會建立以下擴展組態。

```
Metric value

-infinity          30%    40%          60%     70%             infinity
-------------------------------------------------------------------------
          -30%      | -10% | Unchanged  | +10%  |       +30%
-------------------------------------------------------------------------
```

現在，假設您在可擴展的目標上使用這組擴展設定，其原始容量為 10。下列幾點摘要說明與可擴展目標的容量有關之擴展組態的行為：

- 當彙總指標值大於 40 且小於 60 時，將維持原始容量。
- 如果指標值達到 60，Application Auto Scaling 會將可擴展目標的容量增加 1，達到 11。這是根據向內擴展政策的第二步調整 (增加 10 的 10%)。加入了新的容量後，Application Auto Scaling 會將目前的容量增加到 11。此次容量增加後，如果指標值仍升至 70，Application Auto Scaling 會將目標容量增加 3，達到 14。這是根據向外擴展政策的第二步調整 (增加 11 的 30% 即 3.3，無條件捨入到 3)。
- 如果指標值達到 40，Application Auto Scaling 會根據縮減政策的第二步驟調整，將可擴展目標的容量減少 1，達到 13 (減去 14 的 10% 即 1.4，無條件捨入到 1)。此次減少容量後，如果指標值仍降至 30，Application Auto Scaling 會根據縮減政策的第三步驟調整，將目標容量減少 3，達到 10 (減去 13 的 30% 即 3.9，無條件捨入到 3)。

當您為擴展政策指定步進調整時，請注意下列事項：

- 步進調整的範圍不得重疊或有間隙。
- 僅其中一項步進調整的下限可為空值 (負無限大)。若某項步進調整的下限為負值，則必須有一項步進調整的下限為空值。
- 僅其中一項步進調整的上限可為空值 (正無限大)。若某項步進調整的上限為正值，則必須有一項步進調整的上限為空值。
- 同一項步進調整的上限及下限不得皆為空值。
- 如果指標值高於違規閾值，則含下限而不含上限。如果指標值低於違規閾值，則不含下限而含上限。

擴展調整類型

您可以根據選擇的擴展調整類型，定義執行最佳擴展動作的擴展政策。您可以將調整類型指定為可擴展目標的目前容量百分比或以絕對數字指定。

Application Auto Scaling 的步驟擴展政策支援以下調整類型：

- ChangeInCapacity - 將可擴展目標目前的容量增加或減少指定的值。正值即增加容量，負值則減少容量。例如：如果目前容量為 3 而調整值為 5，Application Auto Scaling 會將容量增加 5，總數達到 8。
- ExactCapacity - 將可擴展目標目前的容量變更為指定的值。此調整類型應指定非負值。例如：如果目前容量為 3 而調整值為 5，Application Auto Scaling 會將容量變更為 5。
- PercentChangeInCapacity - 將可擴展目標目前的容量增加或減少指定的百分比。正值即增加容量，負值則減少容量。例如：如果目前容量為 10 而調整值為 10%，Application Auto Scaling 會將容量增加 1，總數達到 11。

  若結果值不是整數，Application Auto Scaling 的四捨五入規則如下所示：
  - 大於 1 的值無條件捨去取整。例如，12.7 捨入到 12。
  - 0 至 1 之間的值捨入到 1。例如，.67 捨入到 1。
  - 0 至 -1 之間的值捨入到 -1。例如，-.58 捨入到 -1。
  - 小於 -1 的值無條件進位取整。例如，-6.67 捨入到 -6。

  使用 PercentChangeInCapacity 時，您還可透過 MinAdjustmentMagnitude 參數指定最小擴展量。例如，假設您建立了一個增加 25% 的政策，並指定最小擴展量為 2。如果可擴展性目標的容量為 4 且執行此擴展政策，4 的 25% 即為 1。不過，由於您指定最小增量為 2，Application Auto Scaling 將增加 2。

冷卻時間

您可以選擇在步數擴展政策中定義冷卻時間。

冷卻時間指定擴展政策等候上一個擴展活動生效的時間量。

有兩種方法可以規劃使用步驟擴展組態的冷卻時間：

- 橫向擴展冷卻時間的目的是連續(但並非過度)規模擴展 。在 Application Auto Scaling 使用擴展政策成功擴展之後，就會開始計算冷卻時間。除非觸發較大的橫向擴展或冷卻時間結束，否則擴展規模政策不會再次增加所需的容量。在向外擴展冷卻期間有效時，由起始冷卻的向外擴展活動所增加的容量，將計入下次向外擴展活動所需的容量。
- 縮減冷卻時間的目的是保守縮減以保護應用程式的可用性，因此縮減冷卻時間到期後才會開放縮減活動。不過，若在向內擴展冷卻期間另有警示觸發了向外擴展活動，Application Auto Scaling 則會立即向外擴展目標。在這種情況下，縮減冷卻時間隨即停止，且不會完成。

例如，當流量尖峰發生時，會觸發警示，而 Application Auto Scaling 會自動增加容量以協助處理增加的負載。如果為橫向擴展政策設定了冷卻時間，則在警示觸發政策以使容量增加 2 時，擴展活動即成功完成並開始計算橫向擴展冷卻期間。在冷卻時間內，如果警示再次觸發但進行更大幅度的步驟調整 (3)，則前次增加的 2 將視為目前容量的一部分。因此，容量只會再增加 1。與等待冷卻時間到期相比，這可以更快地擴展，但不會增加比所需更多的容量。

冷卻期間是以秒為單位進行測量，且僅適用於擴展政策相關擴展活動。在冷卻期間，當已排定的動作在已排定的時間開始時，它可以立即觸發擴展活動，而無須等候冷卻期間過期。

如未指定任何值，則預設值為 300。

建立、管理及刪除擴展政策常用的命令

擴展政策常用的命令包括：

- register-scalable-target 將 AWS 或自訂資源註冊為可擴展的目標 (Application Auto Scaling 可以擴展的資源），以及暫停和繼續擴展。
- put-calable-policy，新增或修改現有可擴展目標的擴展政策。
- describe-scaling-activities，傳回 AWS 區域的擴展活動相關資訊。
- describe-scaling-policies，傳回 AWS 區域的擴展政策相關資訊。
- delete-scaling-policy，刪除擴展政策。

考量事項

使用步進擴展政策時，下列考量適用：

- 考慮您是否能夠準確地預測應用程式上的步進調整，以便使用步進擴展。如果您的擴展指標可按比例提高或降低可擴展目標容量，我們建議您改用目標追蹤擴展政策。您仍然可以選擇使用步進擴展作為其他政策，以進行更進階的設定。例如，您可以設定使用率達到特定層級時更積極的回應。
- 請務必在橫向擴展閾值和縮減閾值之間選擇足夠的邊際，以防止震盪。振盪不穩是指向內縮減和水平擴展無限循環的現象。也就是說，如果採取擴展動作，指標值將會改變，並反向展開另一次擴展動作。

相關資源

如需有關建立 Auto Scaling 群組的步進擴展政策之詳細資訊，請參閱《Amazon EC2 Auto Scaling 使用者指南》中的「Amazon EC2 Auto Scaling 的步進和簡單擴展政策」。

主控台存取

主控台對可擴展資源的檢視、新增、更新或移除步進擴展政策的存取，視您所使用的資源而定。如需詳細資訊，請參閱AWS 服務 可與 Application Auto Scaling 搭配使用。

使用 建立 Application Auto Scaling 的步驟擴展政策 AWS CLI

此範例使用 AWS CLI 命令來建立 Amazon ECS 服務的步進擴展政策。對於不同的可擴展性目標，請在 中指定其命名空間`--service-namespace`、在 中指定其可擴展性維度`--scalable-dimension`，以及在 中指定其資源 ID`--resource-id`。

使用 時 AWS CLI，請記住，您的命令會在為設定檔 AWS 區域 設定的 中執行。如果您想在不同區域中執行命令，則可變更設定檔的預設區域，或搭配 `--region` 參數使用命令。

任務

- 步驟 1：註冊可擴展的目標
- 步驟 2：建立步驟擴展政策
- 步驟 3：建立叫用擴展政策的警示

步驟 1：註冊可擴展的目標

如果您尚未註冊可擴展的目標，請註冊。使用 register-scalable-target 命令，將目標服務中的特定資源註冊為可擴展的目標。下列範例向 Application Auto Scaling 註冊 Amazon ECS 服務。Application Auto Scaling 可擴展的任務數量下限為 2 個任務，上限為 10 個任務。將每個*********替換為自己的資訊。

Linux、macOS 或 Unix

```
aws application-autoscaling register-scalable-target --service-namespace ecs \
  --scalable-dimension ecs:service:DesiredCount \
  --resource-id service/my-cluster/my-service \
  --min-capacity 2 --max-capacity 10
```

Windows

```
aws application-autoscaling register-scalable-target --service-namespace ecs ^
  --scalable-dimension ecs:service:DesiredCount ^
```

```
  --resource-id service/my-cluster/my-service ^
  --min-capacity 2 --max-capacity 10
```

輸出

如果成功，此命令會傳回可擴展目標的 ARN。下列為範例輸出。

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
}
```

步驟 2：建立步驟擴展政策

若要為可擴展的目標建立步驟擴展政策，您可以使用下列範例來協助您開始使用。

Scale out

建立橫向擴展的步驟擴展政策（增加容量）

1. 使用下列cat命令，將步進擴展政策組態存放在主目錄中名為 config.json的 JSON 檔案中。以下是調整類型為 的範例組態PercentChangeInCapacity，可根據下列步驟調整（假設 CloudWatch 警示閾值為 70) 來增加可擴展目標的容量：

   - 當指標的值大於或等於 70 但小於 85 時，將容量增加 10%
   - 當指標的值大於或等於 85 但小於 95 時，將容量增加 20%
   - 當指標的值大於或等於 95 時，將容量增加 30%

   ```
   $ cat ~/config.json
   {
     "AdjustmentType": "PercentChangeInCapacity",
     "MetricAggregationType": "Average",
     "Cooldown": 60,
     "MinAdjustmentMagnitude": 1,
     "StepAdjustments": [
       {
         "MetricIntervalLowerBound": 0.0,
         "MetricIntervalUpperBound": 15.0,
         "ScalingAdjustment": 10
       },
   ```

```
    {
      "MetricIntervalLowerBound": 15.0,
      "MetricIntervalUpperBound": 25.0,
      "ScalingAdjustment": 20
    },
    {
      "MetricIntervalLowerBound": 25.0,
      "ScalingAdjustment": 30
    }
  ]
}
```

如需詳細資訊，請參閱 Application Auto Scaling API 參考中的 StepScalingPolicyConfiguration。

2. 使用以下 put-scaling-policy 命令並指定您建立的 config.json 檔案，建立名為 my-step-scaling-policy 的擴展政策。

Linux、macOS 或 Unix

```
aws application-autoscaling put-scaling-policy --service-namespace ecs \
  --scalable-dimension ecs:service:DesiredCount \
  --resource-id service/my-cluster/my-service \
  --policy-name my-step-scaling-policy --policy-type StepScaling \
  --step-scaling-policy-configuration file://config.json
```

Windows

```
aws application-autoscaling put-scaling-policy --service-namespace ecs ^
  --scalable-dimension ecs:service:DesiredCount ^
  --resource-id service/my-cluster/my-service ^
  --policy-name my-step-scaling-policy --policy-type StepScaling ^
  --step-scaling-policy-configuration file://config.json
```

輸出

該輸出包含 ARN，其作用為政策的唯一名稱。您需要它來為您的政策建立 CloudWatch 警示。下列為範例輸出。

```
{
    "PolicyARN":
 "arn:aws:autoscaling:region:123456789012:scalingPolicy:ac542982-cbeb-4294-891c-
```

```
a5a941dfa787:resource/ecs/service/my-cluster/my-service:policyName/my-step-
scaling-policy"
}
```

Scale in

建立縮減的步驟擴展政策（減少容量）

1. 使用下列cat命令，將步進擴展政策組態存放在主目錄中名為 config.json的 JSON 檔案中。以下是調整類型為 的範例組態ChangeInCapacity，可根據下列步驟調整（假設 CloudWatch 警示閾值為 50) 來減少可擴展目標的容量：

   - 當指標的值小於或等於 50 但大於 40 時，將容量減少 1
   - 當指標的值小於或等於 40 但大於 30 時，將容量減少 2
   - 當指標的值小於或等於 30 時，將容量減少 3

```
$ cat ~/config.json
{
  "AdjustmentType": "ChangeInCapacity",
  "MetricAggregationType": "Average",
  "Cooldown": 60,
  "StepAdjustments": [
    {
      "MetricIntervalUpperBound": 0.0,
      "MetricIntervalLowerBound": -10.0,
      "ScalingAdjustment": -1
    },
    {
      "MetricIntervalUpperBound": -10.0,
      "MetricIntervalLowerBound": -20.0,
      "ScalingAdjustment": -2
    },
    {
      "MetricIntervalUpperBound": -20.0,
      "ScalingAdjustment": -3
    }
  ]
}
```

如需詳細資訊，請參閱 Application Auto Scaling API 參考中的 StepScalingPolicyConfiguration。

2. 使用以下 put-scaling-policy 命令並指定您建立的 config.json 檔案，建立名為 my-step-scaling-policy 的擴展政策。

   Linux、macOS 或 Unix

```
aws application-autoscaling put-scaling-policy --service-namespace ecs \
  --scalable-dimension ecs:service:DesiredCount \
  --resource-id service/my-cluster/my-service \
  --policy-name my-step-scaling-policy --policy-type StepScaling \
  --step-scaling-policy-configuration file://config.json
```

   Windows

```
aws application-autoscaling put-scaling-policy --service-namespace ecs ^
  --scalable-dimension ecs:service:DesiredCount ^
  --resource-id service/my-cluster/my-service ^
  --policy-name my-step-scaling-policy --policy-type StepScaling ^
  --step-scaling-policy-configuration file://config.json
```

   輸出

   該輸出包含 ARN，其作用為政策的唯一名稱。您需要此 ARN 才能為您的政策建立 CloudWatch 警示。下列為範例輸出。

```
{
    "PolicyARN":
 "arn:aws:autoscaling:region:123456789012:scalingPolicy:ac542982-cbeb-4294-891c-a5a941dfa787:resource/ecs/service/my-cluster/my-service:policyName/my-step-scaling-policy"
}
```

步驟 3：建立叫用擴展政策的警示

最後，使用以下 CloudWatch put-metric-alarm 命令建立警示，以搭配步驟擴展政策使用。此範例將根據平均 CPU 使用率觸發警示。如果該警示至少連續兩次在 60 秒的評估期間達到閾值 70%，其將設定

成處於 ALARM 狀態。若要指定不同的 CloudWatch 指標或使用您自己的自訂指標，請以 `--metric-name` 指定名稱，以 `--namespace` 指定命名空間。

Linux、macOS 或 Unix

```
aws cloudwatch put-metric-alarm --alarm-name Step-Scaling-AlarmHigh-ECS:service/my-cluster/my-service \
  --metric-name CPUUtilization --namespace AWS/ECS --statistic Average \
  --period 60 --evaluation-periods 2 --threshold 70 \
  --comparison-operator GreaterThanOrEqualToThreshold \
  --dimensions Name=ClusterName,Value=default Name=ServiceName,Value=sample-app-service \
  --alarm-actions PolicyARN
```

Windows

```
aws cloudwatch put-metric-alarm --alarm-name Step-Scaling-AlarmHigh-ECS:service/my-cluster/my-service ^
  --metric-name CPUUtilization --namespace AWS/ECS --statistic Average ^
  --period 60 --evaluation-periods 2 --threshold 70 ^
  --comparison-operator GreaterThanOrEqualToThreshold ^
  --dimensions Name=ClusterName,Value=default Name=ServiceName,Value=sample-app-service ^
  --alarm-actions PolicyARN
```

使用 描述 Application Auto Scaling 的步驟擴展政策 AWS CLI

您可以使用 describe-scaling-policies 命令來描述服務命名空間的所有擴展政策。下列範例說明所有 Amazon ECS 服務的所有擴展政策。若要列出特定 Amazon ECS 服務的項目，請只新增 `--resource-id`選項。

```
aws application-autoscaling describe-scaling-policies --service-namespace ecs
```

使用 `--query` 參數可將結果篩選為僅包含步驟擴展政策。如需 query 語法的詳細資訊，請參閱《AWS Command Line Interface 使用者指南》中的從 AWS CLI控制命令輸出。

Linux、macOS 或 Unix

```
aws application-autoscaling describe-scaling-policies --service-namespace ecs \
  --query 'ScalingPolicies[?PolicyType==`StepScaling`]'
```

Windows

```
aws application-autoscaling describe-scaling-policies --service-namespace ecs ^
  --query "ScalingPolicies[?PolicyType==`StepScaling`]"
```

輸出

下列為範例輸出。

```
[
    {
        "PolicyARN": "PolicyARN",
        "StepScalingPolicyConfiguration": {
            "MetricAggregationType": "Average",
            "Cooldown": 60,
            "StepAdjustments": [
                {
                    "MetricIntervalLowerBound": 0.0,
                    "MetricIntervalUpperBound": 15.0,
                    "ScalingAdjustment": 1
                },
                {
                    "MetricIntervalLowerBound": 15.0,
                    "MetricIntervalUpperBound": 25.0,
                    "ScalingAdjustment": 2
                },
                {
                    "MetricIntervalLowerBound": 25.0,
                    "ScalingAdjustment": 3
                }
            ],
            "AdjustmentType": "ChangeInCapacity"
        },
        "PolicyType": "StepScaling",
        "ResourceId": "service/my-cluster/my-service",
        "ServiceNamespace": "ecs",
        "Alarms": [
            {
                "AlarmName": "Step-Scaling-AlarmHigh-ECS:service/my-cluster/my-
service",
                "AlarmARN": "arn:aws:cloudwatch:region:012345678910:alarm:Step-Scaling-
AlarmHigh-ECS:service/my-cluster/my-service"
            }
```

```
        ],
        "PolicyName": "my-step-scaling-policy",
        "ScalableDimension": "ecs:service:DesiredCount",
        "CreationTime": 1515024099.901
    }
]
```

使用 刪除 Application Auto Scaling 的步驟擴展政策 AWS CLI

當您不再需要步驟擴展政策時，可以將其刪除。若要同時刪除擴展政策和相關聯的 CloudWatch 警示，請完成下列任務。

刪除擴展政策

使用 delete-scaling-policy 命令。

Linux、macOS 或 Unix

```
aws application-autoscaling delete-scaling-policy --service-namespace ecs \
  --scalable-dimension ecs:service:DesiredCount \
  --resource-id service/my-cluster/my-service \
  --policy-name my-step-scaling-policy
```

Windows

```
aws application-autoscaling delete-scaling-policy --service-namespace ecs ^
  --scalable-dimension ecs:service:DesiredCount ^
  --resource-id service/my-cluster/my-service ^
  --policy-name my-step-scaling-policy
```

刪除 CloudWatch 警示

使用 delete-alarms 命令。您可以同時刪除一或多個警示。例如，使用下列命令來刪除 Step-Scaling-AlarmHigh-ECS:service/my-cluster/my-service 和 Step-Scaling-AlarmLow-ECS:service/my-cluster/my-service 警示。

```
aws cloudwatch delete-alarms --alarm-name Step-Scaling-AlarmHigh-ECS:service/my-cluster/my-service Step-Scaling-AlarmLow-ECS:service/my-cluster/my-service
```

Application Auto Scaling 的預測擴展

預測擴展會主動擴展您的應用程式。預測擴展會分析歷史負載資料，以偵測流量中的每日或每週模式。它使用此資訊來預測未來容量需求，以主動增加應用程式的容量以符合預期的負載。

預測擴展非常適合以下情況：

- 循環流量，例如正常上班時間資源使用量大，夜間和週末資源使用量小
- 經常性on-and-off工作負載模式，例如批次處理、測試或定期資料分析。
- 需要長時間才能初始化的應用程式，會在擴增事件期間對應用程式效能造成明顯的延遲影響

目錄

Application Auto Scaling 預測擴展的運作方式

若要使用預測擴展，請建立預測擴展政策，指定要監控和分析的 CloudWatch 指標。您可以使用預先定義的指標或自訂指標。若要讓預測擴展開始預測未來值，此指標必須至少有 24 小時的資料。

建立政策之後，預測擴展會開始分析過去 14 天內的指標資料，以識別模式。它使用此分析來產生未來 48 小時容量需求的每小時預測。預測會使用最新的 CloudWatch 資料，每 6 小時更新一次。隨著新資料傳入，預測擴展能夠持續改善未來預測的準確性。

您可以先在僅限預測模式下啟用預測擴展。在此模式中，它會產生容量預測，但不會根據這些預測實際擴展您的容量。這可讓您評估預測的準確性和適用性。

在您檢閱預測資料並決定根據該資料開始擴展後，請將擴展政策切換為預測和擴展模式。在此模式中：

- 如果預測預期負載增加，預測擴展將增加容量。
- 如果預測預期負載減少，預測擴展將不會縮減以移除容量。這可確保只有在需求實際下降時才能縮減，而不只是在預測上。若要移除不再需要的容量，您必須建立目標追蹤或步進擴展政策，因為它們會回應即時指標資料。

根據預設，預測擴展會根據該小時的預測，在每小時開始時擴展可擴展的目標。您可以在 PutScalingPolicy API 操作中使用 SchedulingBufferTime 屬性，選擇性地指定較早的開始時間。這可讓您在預測需求之前啟動預測的容量，讓新容量有足夠時間處理流量。

容量上限

根據預設，當設定擴展政策時，它們不能將容量增加到高於其最大容量。

或者，您可以允許在預測容量接近或超過可擴展目標的最大容量時，自動增加可擴展目標的最大容量。若要啟用此行為，請使用 PutScalingPolicy API 操作中的 MaxCapacityBreachBehavior和 MaxCapacityBuffer 屬性，或 中的最大容量行為設定 AWS Management Console。

> **Warning**
>
> 允許自動增加最大容量時請小心。最大容量不會自動減少回原始最大容量。

建立、管理及刪除擴展政策常用的命令

使用預測擴展政策的常用命令包括：

- register-scalable-target 將 AWS 或自訂資源註冊為可擴展的目標、暫停擴展和恢復擴展。
- put-scaling-policy 來建立預測擴展政策。
- get-predictive-scaling-forecast 擷取預測擴展政策的預測資料。
- describe-scaling-activities 傳回 中擴展活動的相關資訊 AWS 區域。
- describe-scaling-policies 傳回 中擴展政策的相關資訊 AWS 區域。
- delete-scaling-policy 刪除擴展政策。

自訂指標

自訂指標可用來預測應用程式所需的容量。當預先定義的指標不足以擷取應用程式的負載時，自訂指標很有用。

考量事項

使用預測擴展時，下列考量適用。

- 確認預測擴展是否適合您的應用程式。如果應用程式顯示特定於星期幾或星期幾的週期性負載模式，則非常適合預測擴展。先評估預測，再讓預測擴展主動擴展您的應用程式。
- 預測擴展需要至少 24 小時的歷史資料才能開始預測。不過，如果歷史資料跨越整整兩週，那麼預測會更加有效。
- 選擇準確代表應用程式完全載入的負載指標，是應用程式最重要的擴展層面。

建立 Application Auto Scaling 的預測擴展政策

下列範例政策使用 AWS CLI 來設定 Amazon ECS 服務的預測擴展政策。將每個*########*替換為自己的資訊。

如需可指定 CloudWatch 指標的詳細資訊，請參閱《Amazon EC2 Auto Scaling API 參考》中的 PredictiveScalingMetricSpecification。

以下是具有預先定義記憶體組態的範例政策。

```
cat policy.json
{
    "MetricSpecifications": [
        {
            "TargetValue": 40,
            "PredefinedMetricPairSpecification": {
                "PredefinedMetricType": "ECSServiceMemoryUtilization"
            }
        }
    ],
    "SchedulingBufferTime": 3600,
    "MaxCapacityBreachBehavior": "HonorMaxCapacity",
    "Mode": "ForecastOnly"
}
```

下列範例說明透過使用指定的組態檔案執行 put-scaling-policy 命令來建立政策。

```
aws aas put-scaling-policy \
--service-namespace ecs \
--region us-east-1 \
--policy-name predictive-scaling-policy-example \
--resource-id service/MyCluster/test \
--policy-type PredictiveScaling \
--scalable-dimension ecs:service:DesiredCount \
```

```
--predictive-scaling-policy-configuration file://policy.json
```

如果成功，此命令會傳回政策的 ARN。

```
{
"PolicyARN": "arn:aws:autoscaling:us-
east-1:012345678912:scalingPolicy:d1d72dfe-5fd3-464f-83cf-824f16cb88b7:resource/ecs/
service/MyCluster/test:policyName/predictive-scaling-policy-example",
"Alarms": []
}
```

使用排程動作覆寫預測值

有時候，您可能會有未來應用程式需求的其他資訊，但預測計算無法考量該資訊。例如，預測計算可能會低估即將到來的行銷活動所需的容量。您可以使用排程動作，在未來時段暫時覆寫預測。排程動作可以週期性執行，或在一次性需求波動的特定日期與時間執行。

例如，您可以建立最小容量高於預測容量的排程動作。在執行時間，Application Auto Scaling 會更新可擴展目標的最低容量。由於預測擴展會針對容量進行最佳化，因此會接受容量下限高於預測值的排程動作。這樣可以防止容量低於預期。若要停止複寫預測，請使用第二個排程動作，讓最小容量恢復至其原始設定。

下列程序概述在未來時段覆寫預測的步驟。

主題

> **Important**
>
> 本主題假設您嘗試覆寫預測，以擴展到比預測更高的容量。如果您需要暫時減少容量，而不會受到預測擴展政策的干擾，請改用預測僅限模式。在僅限預測模式下，預測擴展將繼續產生預測，但不會自動增加容量。然後，您可以監控資源使用率，並視需要手動減少群組的大小。

步驟 1：(選用) 分析時間序列資料

從分析預測時間序列資料開始。這是選用步驟，但是如果您想要了解預測的詳細資訊，這會很有幫助。

1. 擷取預測

   建立預測之後，您可以查詢預測中的特定時段。查詢的目標是取得特定時段的時間序列資料的完整檢視。

   查詢最多可包含未來兩天的預測資料。如果已經使用預測擴展一段時間，您也可以存取過去的預測資料。不過，開始和結束時間之間的最大持續時間是 30 天。

   若要擷取預測，請使用 get-predictive-scaling-forecast 命令。下列範例會取得 Amazon ECS 服務的預測擴展預測。

   ```
   aws application-autoscaling get-predictive-scaling-forecast --service-namespace ecs
    \
       --scalable-dimension ecs:service:DesiredCount \
       --resource-id 1234567890abcdef0
       --policy-name predictive-scaling-policy \
       --start-time "2021-05-19T17:00:00Z" \
       --end-time "2021-05-19T23:00:00Z"
   ```

   回應包含兩個預測： LoadForecast和 CapacityForecast。 LoadForecast 顯示每小時負載預測。 CapacityForecast顯示每小時處理預測負載所需的容量預測值，同時維持指定的 TargetValue。

2. 確定目標時段

   確定應發生一次性需求波動時的一個小時或數個小時。請記住，預測中顯示的日期和時間為 UTC 格式。

步驟 2：建立兩個排程動作

接下來，在應用程式具有高於預測的負載時，為特定時段建立兩個排程動作。舉例來說，如果行銷活動會在特定時段為網站帶來流量，您可以排程一次性動作，在它開始時更新最小容量。然後，排程另一個動作，以便在事件結束時將最小容量恢復至原始設定。

為一次性事件建立兩個排程動作 (AWS CLI)

若要建立排程動作，請使用 put-scheduled-action 命令。

下列範例定義了 Amazon EC2 Auto Scaling 的排程，該排程在 5 月 19 日下午 5：00 維持至少三個執行個體的容量達 8 小時。下列命令顯示如何實作此案例。

第一個 put-scheduled-update-group-action 命令會指示 Amazon EC2 Auto Scaling 在 2021 年 5 月 19 日下午 5 點 (UTC) 更新 Auto Scaling 群組的最小容量。

```
aws autoscaling put-scheduled-update-group-action --scheduled-action-name my-event-
start \
  --auto-scaling-group-name my-asg --start-time "2021-05-19T17:00:00Z" --minimum-
capacity 3
```

第二個命令指示 Amazon EC2 Auto Scaling 在 2021 年 5 月 20 日上午 1 點 (UTC)，將群組的最小容量設定為 1。

```
aws autoscaling put-scheduled-update-group-action --scheduled-action-name my-event-end
 \
  --auto-scaling-group-name my-asg --start-time "2021-05-20T01:00:00Z" --minimum-
capacity 1
```

再將這些排程動作新增到 Auto Scaling 群組後，Amazon EC2 Auto Scaling 會執行下列動作：

- 在 2021 年 5 月 19 日下午 5 點 (UTC)，第一個排程動作會執行。如果群組目前擁有少於 3 個執行個體，群組則會擴增至 3 個執行個體。在此時間和接下來的八個小時內，如果預測容量高於實際容量，或者如果動態擴展政策生效，則 Amazon EC2 Auto Scaling 可以繼續擴增。
- 在 2021 年 5 月 20 日上午 1 點 (UTC)，第二個排程動作會執行。這會在事件結束時將最小容量恢復至原始設定。

根據週期性排程擴展

若要每週覆寫相同時段的預測，請建立兩個排程動作，並使用 Cron 表達式提供時間與日期邏輯。

Cron 表達式格式由 5 個以空格分隔的欄位組成：[分鐘] [小時] [一個月的第幾日] [一年的第幾個月] [一週的第幾日]。欄位可以包含任何允許的數值，包括特殊字元。

例如，以下 Cron 表達式會在每周二上午 6:30 執行動作。使用星號作為萬用字元，以比對欄位的所有數值。

```
30 6 * * 2
```

使用自訂指標的進階預測擴展政策

在預測擴展政策中，您可以使用預先定義或自訂的指標。當預先定義的指標不足以描述您的應用程式負載時，自訂指標非常有用。

使用自訂指標建立預測擴展政策時，您可以指定 提供的其他 CloudWatch 指標 AWS，也可以指定您自己定義和發佈的指標。您也可以使用指標數學，將現有的指標彙總並轉換為 AWS 不會自動追蹤的新時間序列。當您合併資料中的值時 (例如，透過計算新的總和或平均值)，它稱為執行彙總。產生的資料稱為彙總。

下一節包含如何建構政策的 JSON 結構的最佳實務和範例。

主題

最佳實務

以下最佳實務可協助您更有效地使用自訂指標：

- 對於負載指標規格，最有用的指標是代表應用程式負載的指標。
- 擴展指標必須與容量成反比。也就是說，如果可擴展的目標增加，擴展指標應該會減少大約相同的比例。為確保預測擴展按預期進行，負載指標和擴展指標還必須彼此密切關聯。
- 目標使用率必須與擴展指標的類型相符。對於使用 CPU 使用率的政策組態，這是一個目標百分比。對於使用輸送量 (如請求或訊息數量) 的政策組態，這是在任何一分鐘間隔內每個執行個體的請求或訊息的目標數量。
- 如果不遵循這些建議，則時間序列的預測未來值可能不正確。若要驗證資料是否正確，您可以檢視預測值。或者，在建立預測擴展政策後，檢查 GetPredictiveScalingForecast API 呼叫傳回的 `LoadForecast` 和 `CapacityForecast` 物件。
- 強烈建議您在 forecast only (僅預測) 模式中設定預測擴展，以便在預測擴展主動擴展容量之前評估預測。

先決條件

若要在預測擴展政策中新增自訂指標，您必須擁有 `cloudwatch:GetMetricData` 許可。

若要指定自己的指標，而不是 AWS 提供的指標，您必須先將指標發佈至 CloudWatch。如需詳細資訊，請參閱《Amazon CloudWatch 使用者指南》中的發佈自訂指標。

如果您發佈自己的指標，應確保以至少五分鐘的頻率發佈資料點。Application Auto Scaling 會根據所需的期間長度，從 CloudWatch 擷取資料點。例如，負載指標規範使用每小時指標來衡量應用程式的負載。CloudWatch 使用您發佈的指標資料，透過將時間戳記落於同一小時週期內的所有資料點彙總，為任何一小時週期提供單一資料值。

建構自訂指標的 JSON

下一節包含如何設定預測擴展以從 CloudWatch for Amazon EC2 Auto Scaling 查詢資料的範例。設定此選項有兩種不同的方法，而您選擇的方法會影響您用來建構預測擴展政策的 JSON 的格式。使用指標數學時，JSON 的格式會根據所執行的指標數學而進一步變化。

1. 若要建立直接從 AWS 您發佈至 CloudWatch 的其他 CloudWatch 指標取得資料的政策，請參閱 具有自訂負載和擴展指標的預測擴展政策範例 (AWS CLI)。
2. 若要建立能查詢多個 CloudWatch 指標並使用數學表達式根據這些指標來建立新的時間序列的政策，請參閱 使用指標數學表達式。

具有自訂負載和擴展指標的預測擴展政策範例 (AWS CLI)

若要使用 建立具有自訂負載和擴展指標的預測擴展政策 AWS CLI，請將 的引數存放在名為 `--predictive-scaling-configuration`的 JSON 檔案中`config.json`。

您可以藉由將以下範例中的可替換值換成您的指標和目標使用率的值，開始新增自訂指標。

```
{
  "MetricSpecifications": [
    {
      "TargetValue": 50,
      "CustomizedScalingMetricSpecification": {
        "MetricDataQueries": [
          {
            "Id": "scaling_metric",
            "MetricStat": {
              "Metric": {
                "MetricName": "MyUtilizationMetric",
```

```
                "Namespace": "MyNameSpace",
                "Dimensions": [
                  {
                    "Name": "MyOptionalMetricDimensionName",
                    "Value": "MyOptionalMetricDimensionValue"
                  }
                ]
              },
              "Stat": "Average"
            }
          }
        ]
      },
      "CustomizedLoadMetricSpecification": {
        "MetricDataQueries": [
          {
            "Id": "load_metric",
            "MetricStat": {
              "Metric": {
                "MetricName": "MyLoadMetric",
                "Namespace": "MyNameSpace",
                "Dimensions": [
                  {
                    "Name": "MyOptionalMetricDimensionName",
                    "Value": "MyOptionalMetricDimensionValue"
                  }
                ]
              },
              "Stat": "Sum"
            }
          }
        ]
      }
    }
  ]
}
```

如需詳細資訊，請參閱《Amazon EC2 Auto Scaling API 參考》中的 MetricDataQuery。

> **Note**
>
> 以下是一些其他資源，可協助您尋找 CloudWatch 指標的指標名稱、命名空間、維度和統計資料：

- 如需 AWS 服務可用指標的相關資訊，請參閱《Amazon AWS CloudWatch 使用者指南》中的發佈 CloudWatch 指標的 服務。 Amazon CloudWatch
- 若要使用 取得 CloudWatch 指標的確切指標名稱、命名空間和維度 （ 如適用 ） AWS CLI，請參閱 清單指標。

若要建立此政策，執行使用 JSON 檔案作為輸入 put-scaling-policy 命令，如以下範例所示。

```
aws autoscaling put-scaling-policy --policy-name my-predictive-scaling-policy \
  --auto-scaling-group-name my-asg --policy-type PredictiveScaling \
  --predictive-scaling-configuration file://config.json
```

如果成功，此命令會傳回政策的 Amazon Resource Name (ARN)。

```
{
  "PolicyARN": "arn:aws:autoscaling:region:account-id:scalingPolicy:2f4f5048-d8a8-4d14-
b13a-d1905620f345:autoScalingGroupName/my-asg:policyName/my-predictive-scaling-policy",
  "Alarms": []
}
```

使用指標數學表達式

下一節提供預測擴展政策的資訊和範例，說明如何在政策中使用指標數學。

主題

- 了解指標數學
- Amazon EC2 Auto Scaling 的範例預測擴展政策，使用指標數學結合指標 (AWS CLI)
- 藍/綠部署情境中使用的預測擴展政策範例 (AWS CLI)

了解指標數學

如果您只想彙總現有指標資料，則 CloudWatch 指標數學可以節省將另一個指標發佈至 CloudWatch 的工作量和成本。您可以使用 AWS 提供的任何指標，也可以使用您在應用程式中定義的指標。

如需詳細資訊，請參閱《Amazon CloudWatch 使用者指南》中的使用指標數學。

如果選擇在預測擴展政策中使用指標數學表達式，則請考慮以下幾點：

- 指標數學運算會使用指標名稱、命名空間和指標維度鍵/值對之唯一組合的資料點。

- 您可以使用任何算術運算子 (+-*/^)、統計函數 (如 AVG 或 SUM) 或 CloudWatch 支援的其他函數。
- 您可以在數學表達式的公式中同時使用其他數學表達式的指標和結果。
- 您的指標數學表達式可以由不同的彙總組成。但是，最終彙總結果的最佳實務是將 `Average` 用於擴展指標，`Sum` 用於負載指標。
- 在指標規範中使用的任何表達式都必須最終傳回單一的時間序列。

若要使用指標數學，請執行以下操作：

- 選擇一或多個 CloudWatch 指標。然後，建立表達式。如需詳細資訊，請參閱《Amazon CloudWatch 使用者指南》中的使用指標數學。
- 透過使用 CloudWatch 主控台或 CloudWatch GetMetricData API 驗證指標數學表達式是否有效。

Amazon EC2 Auto Scaling 的範例預測擴展政策，使用指標數學結合指標 (AWS CLI)

有時，您可能需要首先以某種方式處理其資料，而不是直接指定指標。例如，您可能有一個從 Amazon SQS 佇列中提取工作的應用程式，並且您可能想要使用佇列中的項目數量作為預測擴展的條件。佇列中的訊息數量並不僅僅定義所需的執行個體數量。因此，需要更多的工作來建立可用於計算每個執行個體待處理項目的指標。

以下是此案例的預測擴展政策範例。它指定了基於 Amazon SQS `ApproximateNumberOfMessagesVisible` 指標的擴展和負載指標，即可從佇列中擷取的訊息數量。它還使用 Amazon EC2 Auto Scaling `GroupInServiceInstances` 指標和數學表達式來計算每個執行個體的待處理項目，以擴展指標。

```
aws autoscaling put-scaling-policy --policy-name my-sqs-custom-metrics-policy \
  --auto-scaling-group-name my-asg --policy-type PredictiveScaling \
  --predictive-scaling-configuration file://config.json
{
  "MetricSpecifications": [
    {
      "TargetValue": 100,
      "CustomizedScalingMetricSpecification": {
        "MetricDataQueries": [
          {
            "Label": "Get the queue size (the number of messages waiting to be
 processed)",
            "Id": "queue_size",
            "MetricStat": {
              "Metric": {
```

```
            "MetricName": "ApproximateNumberOfMessagesVisible",
            "Namespace": "AWS/SQS",
            "Dimensions": [
              {
                "Name": "QueueName",
                "Value": "my-queue"
              }
            ]
          },
          "Stat": "Sum"
        },
        "ReturnData": false
      },
      {
        "Label": "Get the group size (the number of running instances)",
        "Id": "running_capacity",
        "MetricStat": {
          "Metric": {
            "MetricName": "GroupInServiceInstances",
            "Namespace": "AWS/AutoScaling",
            "Dimensions": [
              {
                "Name": "AutoScalingGroupName",
                "Value": "my-asg"
              }
            ]
          },
          "Stat": "Sum"
        },
        "ReturnData": false
      },
      {
        "Label": "Calculate the backlog per instance",
        "Id": "scaling_metric",
        "Expression": "queue_size / running_capacity",
        "ReturnData": true
      }
    ]
  },
  "CustomizedLoadMetricSpecification": {
    "MetricDataQueries": [
      {
        "Id": "load_metric",
        "MetricStat": {
```

```
              "Metric": {
                "MetricName": "ApproximateNumberOfMessagesVisible",
                "Namespace": "AWS/SQS",
                "Dimensions": [
                  {
                    "Name": "QueueName",
                    "Value": "my-queue"
                  }
                ],
              },
              "Stat": "Sum"
            },
            "ReturnData": true
          }
        ]
      }
    }
  ]
}
```

該範例會傳回政策的 ARN。

```
{
  "PolicyARN": "arn:aws:autoscaling:region:account-id:scalingPolicy:2f4f5048-d8a8-4d14-
b13a-d1905620f345:autoScalingGroupName/my-asg:policyName/my-sqs-custom-metrics-policy",
  "Alarms": []
}
```

藍/綠部署情境中使用的預測擴展政策範例 (AWS CLI)

搜尋表達式提供了進階選項，您可以在該選項中查詢多個 Auto Scaling 群組中的指標，並對其執行數學表達式。這對藍/綠部署尤其實用。

> **Note**
>
> 藍/綠部署是一種部署方法，您可以在其中建立兩個獨立但相同的 Auto Scaling 群組。只有一個群組會接收生產流量。使用者流量最初定向至較早的 Auto Scaling 群組 (「藍色」)，而新群組 (「綠色」) 用於測試和評估應用程式或服務的新版本。測試並接受新部署後，使用者流量將轉移至綠色 Auto Scaling 群組。然後，您可以在部署成功後刪除藍色群組。

當新 Auto Scaling 群組作為藍/綠色部署的一部分建立時，可以自動將每個群組的指標歷史記錄包含在預測擴展政策中，而無需變更其指標規範。如需詳細資訊，請參閱 AWS 運算部落格上的使用 EC2 Auto Scaling 預測擴展政策搭配藍/綠部署。

以下範例政策示範如何執行此操作。在此範例中，政策使用 Amazon EC2 發出的 CPUUtilization 指標。它使用 Amazon EC2 Auto Scaling GroupInServiceInstances 指標和數學表達式來計算每個執行個體的擴展指標值。它還指定了容量指標規範以取得 GroupInServiceInstances 指標。

搜尋表達式會根據指定的搜尋條件尋找多個 Auto Scaling 群組中執行個體的 CPUUtilization。如果您稍後建立了與相同搜尋條件相符的新 Auto Scaling 群組，則會自動包含新 Auto Scaling 群組中執行個體的 CPUUtilization。

```
aws autoscaling put-scaling-policy --policy-name my-blue-green-predictive-scaling-
policy \
  --auto-scaling-group-name my-asg --policy-type PredictiveScaling \
  --predictive-scaling-configuration file://config.json
{
  "MetricSpecifications": [
    {
      "TargetValue": 25,
      "CustomizedScalingMetricSpecification": {
        "MetricDataQueries": [
          {
            "Id": "load_sum",
            "Expression": "SUM(SEARCH('{AWS/EC2,AutoScalingGroupName} MetricName=
\"CPUUtilization\" ASG-myapp', 'Sum', 300))",
            "ReturnData": false
          },
          {
            "Id": "capacity_sum",
            "Expression": "SUM(SEARCH('{AWS/AutoScaling,AutoScalingGroupName}
 MetricName=\"GroupInServiceInstances\" ASG-myapp', 'Average', 300))",
            "ReturnData": false
          },
          {
            "Id": "weighted_average",
            "Expression": "load_sum / capacity_sum",
            "ReturnData": true
          }
        ]
      },
      "CustomizedLoadMetricSpecification": {
        "MetricDataQueries": [
```

```
          {
            "Id": "load_sum",
            "Expression": "SUM(SEARCH('{AWS/EC2,AutoScalingGroupName} MetricName=
\"CPUUtilization\" ASG-myapp', 'Sum', 3600))"
          }
        ]
      },
      "CustomizedCapacityMetricSpecification": {
        "MetricDataQueries": [
          {
            "Id": "capacity_sum",
            "Expression": "SUM(SEARCH('{AWS/AutoScaling,AutoScalingGroupName}
 MetricName=\"GroupInServiceInstances\" ASG-myapp', 'Average', 300))"
          }
        ]
      }
    }
  ]
}
```

該範例會傳回政策的 ARN。

```
{
  "PolicyARN": "arn:aws:autoscaling:region:account-id:scalingPolicy:2f4f5048-d8a8-4d14-
b13a-d1905620f345:autoScalingGroupName/my-asg:policyName/my-blue-green-predictive-
scaling-policy",
  "Alarms": []
}
```

預測擴展政策中自訂指標的考量

如果使用自訂指標時出現問題，建議您執行以下操作：

- 如果提供了錯誤訊息，則請閱讀該訊息，並在可行的狀況下解決其報告的問題。
- 如果您未事先驗證表達式， put-scaling-policy 命令會在您建立擴展政策時驗證它。但是，此命令可能無法識別偵測到之錯誤的確切原因。若要修正這些問題，請對您收到之 get-metric-data 命令請求回應中的錯誤進行疑難排解。您也可以從 CloudWatch 主控台對表達式進行疑難排解。
- 如果 `MetricDataQueries` 自己指定了 SEARCH() 函數 (在無需 SUM() 等數學函數的狀況下)，則您必須為 `ReturnData` 指定 `false`。這是因為搜尋表達式可能會傳回多個時間序列，並且基於表達式的指標規範只能傳回一個時間序列。
- 搜尋表達式中涉及的所有指標均應具有相同的解析度。

限制

具有下列限制。

- 您可以在一個指標規範中查詢最多 10 個指標的資料點。
- 在此限制之下，一個表達式計為一個指標。

教學課程：設定自動擴展以處理繁重的工作負載

在本教學課程中，您將學習如何在應用程式高於正常工作負載的時段水平擴展和縮減。當應用程式可能定期或隨季節而突然訪客大增時，這很有用。

您可以使用目標追蹤擴展政策連同排程擴展功能，以處理額外的負載。排程擴展功能會根據您指定的排程，自動代表您對 `MinCapacity` 和 `MaxCapacity` 起始變更。資源上處於作用中的目標追蹤擴展政策可以根據目前的資源使用率，在新的容量下限和上限範圍內動態擴展。

完成本教學課程後，您將會知道如何：

- 使用排程擴充功能來事先增加額外容量，以因應繁重的負載，等到不需要額外容量時再移除。
- 使用目標追蹤擴展政策以根據目前的資源使用率來擴展應用程式。

目錄

先決條件

此教學假設您已執行下列作業：

- 已建立 AWS 帳戶。
- 安裝並設定 AWS CLI。
- 授予使用 Application Auto Scaling 將資源註冊和取消註冊為可擴展目標的必要許可。此外， 授予建立擴展政策和排程動作的必要許可。如需詳細資訊，請參閱適用於 Application Auto Scaling 的 Identity and Access Management。
- 在可用於本教學課程的非生產環境中建立支援的資源。如果您還沒有，請立即建立一個。如果需要 Application Auto Scaling 可使用的 AWS 服務和資源的相關資訊，請參閱 AWS 服務 可與 Application Auto Scaling 搭配使用 區段。

> **Note**
>
> 完成本教學課程時，有兩個步驟可讓您將資源的最小和最大容量值設定為 0，使目前容量重設為 0。視您使用的 Application Auto Scaling 資源而定，在這些步驟期間，您可能無法將目前容量重設為 0。為了協助您解決問題，輸出中的訊息會指出最小容量不能小於指定的值，並提供 AWS 資源可接受的最小容量值。

步驟 1：註冊可擴展的目標

首先向 Application Auto Scaling 將資源註冊為可擴展的目標。可擴展的目標是 Application Auto Scaling 可水平擴展和縮減的資源。

向 Application Auto Scaling 註冊可擴展的目標

- 使用以下 register-scalable-target 命令註冊可擴展的目標。將 `--min-capacity` 和 `--max-capacity` 值設定為 0，使目前容量重設為 0。

  替換您正在使用的 Application Auto Scaling AWS 服務的命名空間的 `--service-namespace` 範例文本、您正在註冊的資源相關的可擴展維度的 `--scalable-dimension`，和具有資源辨識符的 `--resource-id`。這些值根據使用的資源以及資源 ID 的構建方式而定。如果需要更多相關資訊，請參閱主題內的 AWS 服務 可與 Application Auto Scaling 搭配使用 區段。這些主題包括範例命令，能引導您了解如何使用 Application Auto Scaling 註冊可擴展目標。

  Linux、macOS 或 Unix

  ```
  aws application-autoscaling register-scalable-target \
    --service-namespace namespace \
    --scalable-dimension dimension \
    --resource-id identifier \
    --min-capacity 0 --max-capacity 0
  ```

  Windows

  ```
  aws application-autoscaling register-scalable-target --service-namespace namespace
   --scalable-dimension dimension --resource-id identifier --min-capacity 0 --max-
  capacity 0
  ```

  如果成功，此命令會傳回可擴展目標的 ARN。

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-
id:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
}
```

步驟 2：根據您的需求設定排定的動作

您可以使用 put-scheduled-action 命令建立排定的動作，並設定為滿足您的業務需求。在本教學課程中，我們著重的組態會在工作時間以外將容量降低至 0，以免耗用資源。

建立在早晨水平擴展的排定動作

1. 若要水平擴展可擴展的目標，請使用以下 put-scheduled-action 命令。包括 `--schedule` 參數，並使用 Cron 運算式以 UTC 格式指定週期性排程。

   Application Auto Scaling 會依指定的排程 (每天上午 9 點 UTC)，將 `MinCapacity` 和 `MaxCapacity` 值更新為 1-5 個容量單位的所需範圍。

   Linux、macOS 或 Unix

   ```
   aws application-autoscaling put-scheduled-action \
     --service-namespace namespace \
     --scalable-dimension dimension \
     --resource-id identifier \
     --scheduled-action-name my-first-scheduled-action \
     --schedule "cron(0 9 * * ? *)" \
     --scalable-target-action MinCapacity=1,MaxCapacity=5
   ```

   Windows

   ```
   aws application-autoscaling put-scheduled-action --service-namespace namespace --
   scalable-dimension dimension --resource-id identifier --scheduled-action-name my-
   first-scheduled-action --schedule "cron(0 9 * * ? *)" --scalable-target-action
    MinCapacity=1,MaxCapacity=5
   ```

   如果成功，此命令不會傳回任何輸出。

2. 若要確認排定的動作存在，請使用下列 describe-scheduled-actions 命令。

Linux、macOS 或 Unix

```
aws application-autoscaling describe-scheduled-actions \
  --service-namespace namespace \
  --query 'ScheduledActions[?ResourceId==`identifier`]'
```

Windows

```
aws application-autoscaling describe-scheduled-actions --service-
namespace namespace --query "ScheduledActions[?ResourceId==`identifier`]"
```

下列為範例輸出。

```
[
    {
        "ScheduledActionName": "my-first-scheduled-action",
        "ScheduledActionARN": "arn",
        "Schedule": "cron(0 9 * * ? *)",
        "ScalableTargetAction": {
            "MinCapacity": 1,
            "MaxCapacity": 5
        },
        ...
    }
]
```

建立在夜間水平擴展的排定動作

1. 重複上述程序建立另一個排定的動作，供 Application Auto Scaling 在一天結束時用來縮減。

   Application Auto Scaling 會依指定的排程 (每天下午 8 點 UTC)，將目標的 `MinCapacity` 和 `MaxCapacity` 更新為 0，如以下 put-scheduled-action 命令所指示。

   Linux、macOS 或 Unix

   ```
   aws application-autoscaling put-scheduled-action \
     --service-namespace namespace \
     --scalable-dimension dimension \
     --resource-id identifier \
   ```

```
  --scheduled-action-name my-second-scheduled-action \
  --schedule "cron(0 20 * * ? *)" \
  --scalable-target-action MinCapacity=0,MaxCapacity=0
```

Windows

```
aws application-autoscaling put-scheduled-action --service-namespace namespace --
scalable-dimension dimension --resource-id identifier --scheduled-action-name my-
second-scheduled-action --schedule "cron(0 20 * * ? *)" --scalable-target-action
 MinCapacity=0,MaxCapacity=0
```

2. 若要確認排定的動作存在，請使用下列 describe-scheduled-actions 命令。

Linux、macOS 或 Unix

```
aws application-autoscaling describe-scheduled-actions \
  --service-namespace namespace \
  --query 'ScheduledActions[?ResourceId==`identifier`]'
```

Windows

```
aws application-autoscaling describe-scheduled-actions --service-
namespace namespace --query "ScheduledActions[?ResourceId==`identifier`]"
```

下列為範例輸出。

```
[
    {
        "ScheduledActionName": "my-first-scheduled-action",
        "ScheduledActionARN": "arn",
        "Schedule": "cron(0 9 * * ? *)",
        "ScalableTargetAction": {
            "MinCapacity": 1,
            "MaxCapacity": 5
        },
        ...
    },
    {
        "ScheduledActionName": "my-second-scheduled-action",
        "ScheduledActionARN": "arn",
        "Schedule": "cron(0 20 * * ? *)",
```

```
        "ScalableTargetAction": {
            "MinCapacity": 0,
            "MaxCapacity": 0
        },
        ...
    }
]
```

步驟 3：新增目標追蹤擴展政策

現在您已備妥基本排程，請新增目標追蹤擴展政策，以根據目前的資源使用率來擴展。

Application Auto Scaling 會透過目標追蹤，比較政策中的目標值與特定指標的目前值。當這些值在一段時間內不相等時，Application Auto Scaling 會增加或移除容量，以維持穩定的效能。當應用程式的負載和指標值上升時，Application Auto Scaling 會盡快增加容量，但不會超過 `MaxCapacity`。當 Application Auto Scaling 因負載極小而刪除容量時，不會低於 `MinCapacity`。根據使用量調整容量，您只須為應用程式所需而付費。

如果因為應用程式沒有任何負載，以致指標沒有足夠的資料，則 Application Auto Scaling 不會增加或移除容量。換句話說，Application Auto Scaling 會在資訊不足的情況下，以可用性為優先。

您可以新增多個擴展政策，但絕不可新增衝突的步驟擴展政策，否則可能造成不良後果。例如，如果步進擴展政策在目標追蹤政策準備好縮減之前即啟動縮減活動，則不會封鎖縮減活動。縮減活動完成之後，目標追蹤政策可以指示 Application Auto Scaling 再次水平擴展。

建立目標追蹤擴展政策

1. 使用以下 put-scaling-policy 命令建立政策。

   目標追蹤最常用的指標已預先定義，可以直接使用，不需要從 CloudWatch 提供完整的指標規格。關於有哪些預先定義的指標可用，詳情請參閱 Application Auto Scaling 的目標追蹤擴展政策。

   執行此命令之前，請確定預先定義的指標期待目標值。例如，若要在 CPU 使用率達到 50% 時水平擴展，請指定目標值 50.0。或者，若要在使用率達到 70% 時水平擴展 Lambda 佈建並行，請指定目標值 0.7。關於特定資源的目標值，相關資訊請參閱服務提供的文件，以了解如何設定目標追蹤。如需詳細資訊，請參閱AWS 服務 可與 Application Auto Scaling 搭配使用。

   Linux、macOS 或 Unix

   ```
   aws application-autoscaling put-scaling-policy \
   ```

```
  --service-namespace namespace \
  --scalable-dimension dimension \
  --resource-id identifier \
  --policy-name my-scaling-policy --policy-type TargetTrackingScaling \
  --target-tracking-scaling-policy-configuration '{ "TargetValue": 50.0,
 "PredefinedMetricSpecification": { "PredefinedMetricType": "predefinedmetric" }}'
```

Windows

```
aws application-autoscaling put-scaling-policy --service-namespace namespace --
scalable-dimension dimension --resource-id identifier --policy-name my-scaling-
policy --policy-type TargetTrackingScaling --target-tracking-scaling-policy-
configuration "{ \"TargetValue\": 50.0, \"PredefinedMetricSpecification\":
 { \"PredefinedMetricType\": \"predefinedmetric\" }}"
```

如果成功，此命令會傳回替您建立的兩個 CloudWatch 警示的 ARN 和名稱。

2. 若要確認排定的動作存在，請使用下列 describe-scaling-policies 命令。

Linux、macOS 或 Unix

```
aws application-autoscaling describe-scaling-policies --service-namespace namespace
 \
  --query 'ScalingPolicies[?ResourceId==`identifier`]'
```

Windows

```
aws application-autoscaling describe-scaling-policies --service-namespace namespace
 --query "ScalingPolicies[?ResourceId==`identifier`]"
```

下列為範例輸出。

```
[
    {
        "PolicyARN": "arn",
        "TargetTrackingScalingPolicyConfiguration": {
            "PredefinedMetricSpecification": {
                "PredefinedMetricType": "predefinedmetric"
            },
            "TargetValue": 50.0
        },
```

```
        "PolicyName": "my-scaling-policy",
        "PolicyType": "TargetTrackingScaling",
        "Alarms": [],
        ...
    }
]
```

步驟 4：後續步驟

當擴展活動發生時，您會在可擴展目標的擴展活動輸出值中看到該活動的記錄，例如：

```
Successfully set desired count to 1. Change successfully fulfilled by ecs.
```

若要使用 Application Auto Scaling 來監控擴展活動，您可以使用以下的 describe-scaling-activities 命令。

Linux、macOS 或 Unix

```
aws application-autoscaling describe-scaling-activities
  --service-namespace namespace \
  --scalable-dimension dimension \
  --resource-id identifier
```

Windows

```
aws application-autoscaling describe-scaling-activities --service-namespace namespace
 --scalable-dimension dimension --resource-id identifier
```

步驟 5：清除

若要避免帳戶因為積極擴展時建立的資源而產生費用，您可以按照下列方式清除相關的擴展組態。

刪除擴展組態不會刪除基礎 AWS 資源。也不會恢復到原來的容量。在您建立資源的服務主控台上，您可以刪除資源或調整其容量。

刪除排程動作

以下 delete-scheduled-action 命令會刪除指定的排定動作。如果想要保留您建立的排定動作，您可以略過此步驟。

Linux、macOS 或 Unix

```
aws application-autoscaling delete-scheduled-action \
  --service-namespace namespace \
  --scalable-dimension dimension \
  --resource-id identifier \
  --scheduled-action-name my-second-scheduled-action
```

Windows

```
aws application-autoscaling delete-scheduled-action --service-namespace namespace
 --scalable-dimension dimension --resource-id identifier --scheduled-action-name my-
second-scheduled-action
```

刪除擴展政策

以下 delete-scaling-policy 命令會刪除指定的目標追蹤擴展政策。如果想要保留您建立的擴展政策，您可以略過此步驟。

Linux、macOS 或 Unix

```
aws application-autoscaling delete-scaling-policy \
  --service-namespace namespace \
  --scalable-dimension dimension \
  --resource-id identifier \
  --policy-name my-scaling-policy
```

Windows

```
aws application-autoscaling delete-scaling-policy --service-namespace namespace --
scalable-dimension dimension --resource-id identifier --policy-name my-scaling-policy
```

解除登錄可擴展的目標

使用以下 deregister-scalable-target 命令取消註冊可擴展的目標。如果您有任何由您建立的擴展政策或任何排程動作尚未刪除，此命令會將其全部刪除。如果您想要保留可擴展的目標以供日後使用，您可以略過此步驟。

Linux、macOS 或 Unix

```
aws application-autoscaling deregister-scalable-target \
```

```
  --service-namespace namespace \
  --scalable-dimension dimension \
  --resource-id identifier
```

Windows

```
aws application-autoscaling deregister-scalable-target --service-namespace namespace --
scalable-dimension dimension --resource-id identifier
```

Application Auto Scaling 暫停和繼續擴展

此主題說明如何暫停，然後恢復您應用程式中一或多個可擴展目標的擴展活動。暫停繼續恢復功能可用來暫時暫停透過您的擴展政策和排程動作觸發的擴展活動。這將非常實用，例如當您在進行變更或調查組態問題，而不希望自動擴展產生潛在干擾時。您的擴展政策和排程動作皆可以保留，且在您準備好時可以繼續擴展活動。

在以下範例 CLI 命令中，您在 config.json 檔案中傳遞 JSON 格式的參數。您也可以在命令列以引號括住 JSON 資料結構來傳遞這些參數。如需詳細資訊，請參閱《AWS Command Line Interface 使用者指南》中的在 AWS CLI中使用引號括住字串。

目錄

> **Note**
>
> 如需在 Amazon ECS 部署進行時暫停橫向擴展程序的說明，請參閱下列文件：《Amazon Elastic Container Service 開發人員指南》中的服務自動擴展和部署

擴展活動

Application Auto Scaling 支援將以下擴展活動置於暫停狀態：

- 所有由擴展政策觸發的向內擴展活動。
- 所有由擴展政策觸發的向外擴展活動。
- 所有涉及排程動作的擴展活動。

以下描述說明在個別擴展活動暫停時將發生的事。每一個都可以獨立暫停和恢復。根據擴展活動暫停的原因，您可能需要一起暫停多個擴展活動。

`DynamicScalingInSuspended`

- 觸發目標追蹤擴展政策或步驟擴展政策時，Application Auto Scaling 不會移除容量。這可讓您暫時停用與擴展政策相關聯的縮減活動，但不會刪除擴展政策或其相關聯的 CloudWatch 警示。當您繼續縮減時，Application Auto Scaling 會以目前違反的警示閾值評估政策。

`DynamicScalingOutSuspended`

- 觸發目標追蹤擴展政策或步驟擴展政策時，Application Auto Scaling 不會增加容量。這可讓您暫時停用與擴展政策相關聯的縮減活動，但不會刪除擴展政策或其相關聯的 CloudWatch 警示。當您繼續水平擴展時，Application Auto Scaling 會以目前違反的警示閾值評估政策。

`ScheduledScalingSuspended`

- Application Auto Scaling 不會啟動在暫停期間排定執行的擴展動作。當您繼續排定的擴展時，Application Auto Scaling 只會評估還沒超過執行時間的排定動作。

暫停和繼續擴展活動

您可以為 Application Auto Scaling 可擴展目標暫停和繼續個別擴展活動或所有擴展活動。

> **Note**
>
> 為求簡便，這些範例說明如何暫停和繼續 DynamoDB 資料表的擴展。若要指定不同的可擴展性目標，請以 `--service-namespace` 指定其命名空間，以 `--scalable-dimension` 指定其可擴展維度，以 `--resource-id` 指定其資源 ID。 如需每個服務的詳細資訊和範例，請參閱 AWS 服務 可與 Application Auto Scaling 搭配使用 中的主題。

暫停擴展活動

開啟命令列視窗並使用 register-scalable-target 命令與 `--suspended-state` 選項，如下所示。

Linux、macOS 或 Unix

```
aws application-autoscaling register-scalable-target --service-namespace dynamodb \
  --scalable-dimension dynamodb:table:ReadCapacityUnits --resource-id table/my-table \
  --suspended-state file://config.json
```

Windows

```
aws application-autoscaling register-scalable-target --service-namespace dynamodb --
scalable-dimension dynamodb:table:ReadCapacityUnits --resource-id table/my-table --
suspended-state file://config.json
```

如果成功，此命令會傳回可擴展目標的 ARN。

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-
target/1234abcd56ab78cd901ef1234567890ab123"
}
```

若要僅暫停由擴展政策觸發的向內擴展活動，請在 config.json 中指定以下項目。

```
{
    "DynamicScalingInSuspended":true
}
```

若要僅暫停由擴展政策觸發的向外擴展活動，請在 config.json 中指定以下項目。

```
{
    "DynamicScalingOutSuspended":true
}
```

若要只暫停與排程動作相關的擴展活動，請在 config.json 中指定以下項目。

```
{
    "ScheduledScalingSuspended":true
}
```

暫停所有擴展活動

使用 register-scalable-target 命令與 `--suspended-state` 選項，如下所示。

Linux、macOS 或 Unix

```
aws application-autoscaling register-scalable-target --service-namespace dynamodb \
  --scalable-dimension dynamodb:table:ReadCapacityUnits --resource-id table/my-table \
  --suspended-state file://config.json
```

Windows

```
aws application-autoscaling register-scalable-target --service-namespace dynamodb --
scalable-dimension dynamodb:table:ReadCapacityUnits --resource-id table/my-table --
suspended-state file://config.json
```

此範例假設檔案 config.json 包含下列 JSON 格式的參數。

```
{
    "DynamicScalingInSuspended":true,
    "DynamicScalingOutSuspended":true,
    "ScheduledScalingSuspended":true
}
```

如果成功，此命令會傳回可擴展目標的 ARN。

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-
target/1234abcd56ab78cd901ef1234567890ab123"
}
```

檢視暫停的擴展活動

使用 describe-scalable-targets 命令來判斷可擴展目標的哪些擴展活動處於暫停狀態。

Linux、macOS 或 Unix

```
aws application-autoscaling describe-scalable-targets --service-namespace dynamodb \
  --scalable-dimension dynamodb:table:ReadCapacityUnits --resource-id table/my-table
```

Windows

```
aws application-autoscaling describe-scalable-targets --service-namespace dynamodb --
scalable-dimension dynamodb:table:ReadCapacityUnits --resource-id table/my-table
```

下列為範例輸出。

```
{
    "ScalableTargets": [
        {
            "ServiceNamespace": "dynamodb",
            "ScalableDimension": "dynamodb:table:ReadCapacityUnits",
```

```
            "ResourceId": "table/my-table",
            "MinCapacity": 1,
            "MaxCapacity": 20,
            "SuspendedState": {
                "DynamicScalingOutSuspended": true,
                "DynamicScalingInSuspended": true,
                "ScheduledScalingSuspended": true
            },
            "CreationTime": 1558125758.957,
            "RoleARN": "arn:aws:iam::123456789012:role/aws-
service-role/dynamodb.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_DynamoDBTable"
        }
    ]
}
```

繼續擴展活動

準備要繼續擴展活動時，您可以繼續使用 register-scalable-target 命令來繼續。

以下範例命令會恢復所有指定可擴展目標的擴展活動。

Linux、macOS 或 Unix

```
aws application-autoscaling register-scalable-target --service-namespace dynamodb \
  --scalable-dimension dynamodb:table:ReadCapacityUnits --resource-id table/my-table \
  --suspended-state file://config.json
```

Windows

```
aws application-autoscaling register-scalable-target --service-namespace dynamodb --
scalable-dimension dynamodb:table:ReadCapacityUnits --resource-id table/my-table --
suspended-state file://config.json
```

此範例假設檔案 config.json 包含下列 JSON 格式的參數。

```
{
    "DynamicScalingInSuspended":false,
    "DynamicScalingOutSuspended":false,
    "ScheduledScalingSuspended":false
}
```

如果成功，此命令會傳回可擴展目標的 ARN。

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-
target/1234abcd56ab78cd901ef1234567890ab123"
}
```

Application Auto Scaling 擴展活動

Application Auto Scaling 會監控擴展政策的 CloudWatch 指標，並在超出閾值時啟動擴展活動。手動或按照排程修改可擴展目標的大小上下限時，Application Auto Scaling 也會啟動擴展活動。

進行擴展活動時，Application Auto Scaling 會執行下列其中一項動作：

- 增加可擴展目標的容量 (稱為水平擴展)
- 減少可擴展目標的容量 (稱為向內縮減)

您可以查看過去六週的擴展活動。

依可擴展的目標查詢擴展活動

若要查看特定可擴展目標的擴展活動，請使用 describe-scaling-activities 命令。

Linux、macOS 或 Unix

```
aws application-autoscaling describe-scaling-activities --service-namespace ecs \
  --scalable-dimension ecs:service:DesiredCount --resource-id service/my-cluster/my-service
```

Windows

```
aws application-autoscaling describe-scaling-activities --service-namespace ecs --
scalable-dimension ecs:service:DesiredCount --resource-id service/my-cluster/my-service
```

以下是範例回應，其中 StatusCode 包含目前活動的狀態，而 StatusMessage 則包含擴展活動狀態的相關資訊。

```
{
    "ScalingActivities": [
        {
            "ScalableDimension": "ecs:service:DesiredCount",
            "Description": "Setting desired count to 1.",
            "ResourceId": "service/my-cluster/my-service",
            "ActivityId": "e6c5f7d1-dbbb-4a3f-89b2-51f33e766399",
```

```
            "StartTime": 1462575838.171,
            "ServiceNamespace": "ecs",
            "EndTime": 1462575872.111,
            "Cause": "monitor alarm web-app-cpu-lt-25 in state ALARM triggered policy
 web-app-cpu-lt-25",
            "StatusMessage": "Successfully set desired count to 1. Change successfully
 fulfilled by ecs.",
            "StatusCode": "Successful"
        }
    ]
}
```

如需回應中各欄位的說明，請參閱 Application Auto Scaling API 參考資料中的擴展活動。

下列狀態代碼表示擴展事件觸發擴展活動後，達到完成狀態的時間：

- `Successful` – 擴展已成功完成
- `Overridden` – 所需容量已由較新的擴展事件完成更新
- `Unfulfilled` – 擴展逾時，或目標服務無法履行請求
- `Failed` – 擴展失敗，發生異常狀況

> **Note**
>
> 擴展活動的狀態也可能為 `Pending` 或 `InProgress`。所有擴展活動在目標服務回應之前皆為 `Pending` 狀態。目標回應後，擴展活動的狀態會變更為 `InProgress`。

包含未擴展的活動

預設情況下，擴展活動不會反映 Application Auto Scaling 決定是否要擴展的時間。

舉例來說，假設 Amazon ECS 服務超過指定指標的最大閾值，但任務數量已達到允許的任務數量上限。在此情況下，Application Auto Scaling 不會橫向擴展所需的任務數量。

若要在回應中加入未擴展 (未擴展活動) 的活動，請將 `--include-not-scaled-activities` 選項新增至 describe-scaling-activities 命令。

Linux、macOS 或 Unix

```
aws application-autoscaling describe-scaling-activities --include-not-scaled-activities
 \
  --service-namespace ecs --scalable-dimension ecs:service:DesiredCount \
  --resource-id service/my-cluster/my-service
```

Windows

```
aws application-autoscaling describe-scaling-activities --include-not-scaled-activities
 --service-namespace ecs --scalable-dimension ecs:service:DesiredCount --resource-
id service/my-cluster/my-service
```

> **Note**
>
> 如果此命令擲回錯誤，請確定您已在 AWS CLI 本機將 更新至最新版本。

為確認回應是否含有未擴展的活動，NotScaledReasons 元素會顯示在部分 (若非全部) 失敗擴展活動的輸出中。

```
{
    "ScalingActivities": [
        {
            "ScalableDimension": "ecs:service:DesiredCount",
            "Description": "Attempting to scale due to alarm triggered",
            "ResourceId": "service/my-cluster/my-service",
            "ActivityId": "4d759079-a31f-4d0c-8468-504c56e2eecf",
            "StartTime": 1664928867.915,
            "ServiceNamespace": "ecs",
            "Cause": "monitor alarm web-app-cpu-gt-75 in state ALARM triggered policy
 web-app-cpu-gt-75",
            "StatusCode": "Failed",
            "NotScaledReasons": [
                {
                    "Code": "AlreadyAtMaxCapacity",
                    "MaxCapacity": 4
                }
            ]
        }
    ]
}
```

如需回應中各欄位的說明，請參閱 Application Auto Scaling API 參考資料中的擴展活動。

若系統傳回未擴展的活動，`CurrentCapacity`、`MaxCapacity` 及 `MinCapacity` 等屬性可能會出現在回應中 (視 `Code` 所列的原因代碼而定)。

為了防止大量重複項目，擴展活動歷史記錄中只會記錄第一個未擴展的活動。任何後續未擴展的活動都不會產生新的項目，除非沒有擴展變更的原因。

原因代碼

以下是未擴展活動的原因代碼。

原因代碼	定義			
AutoScalingAnticipatedFlapping	自動擴展演算法決定不進行擴展，因為擴展會導致振盪不穩。振盪不穩是指向內縮減和水平擴展無限循環的現象。也就是說，如果採取擴展動作，指標值將會改變，以反向展開另一次擴展動作。			
TargetServicePutResourceAsUnscalable	目標服務暫時將資源置於無法擴展的狀態。當符合擴展政策中指定的自動擴展條件時，Application Auto Scaling 會嘗試再次擴展。			

原因代碼	定義			
AlreadyAtMaxCapacity	您指定的最大容量已封鎖擴展動作。若您希望 Application Auto Scaling 進行橫向擴展，需提高最大容量。			
AlreadyAtMinCapacity	您指定的最小容量已封鎖擴展動作。若您希望 Application Auto Scaling 進行向內縮減，需降低最大容量。			
AlreadyAtDesiredCapacity	自動擴展演算法計算的修訂容量等於目前的容量。			

監控 Application Auto Scaling

監控是維護 Application Auto Scaling 和其他 AWS 解決方案可靠性、可用性和效能的重要部分。您應該從 AWS 解決方案的所有部分收集監控資料，以便在發生多點失敗時更輕鬆地偵錯。 AWS 提供監控工具來監看 Application Auto Scaling、在發生錯誤時回報，並適時採取自動動作。

您可以使用下列功能來協助您管理 AWS 資源：

AWS CloudTrail

使用 時 AWS CloudTrail，您可以追蹤由 或代表您對 Application Auto Scaling API 發出的呼叫 AWS 帳戶。CloudTrail 會將資訊存放到您所指定的 Amazon S3 儲存貯體的日誌檔案中。您可以找出哪些使用者和帳戶呼叫 Application Auto Scaling、發出呼叫的來源 IP 地址，以及呼叫的發生時間。如需詳細資訊，請參閱使用 記錄 Application Auto Scaling API 呼叫 AWS CloudTrail。

> **Note**
>
> 如需可協助您記錄和收集工作負載相關資料的其他 AWS 服務相關資訊，請參閱《 方案指南》中的應用程式擁有者的記錄和監控指南。 AWS

Amazon CloudWatch

Amazon CloudWatch 可協助您分析日誌，並即時監控 AWS 資源與託管應用程式的指標。您可以收集和追蹤指標、建立自訂儀板表，以及設定警示，在特定指標達到您指定的閾值時通知您或採取動作。例如，您可以讓 CloudWatch 追蹤資源使用率，並在使用率極高或指標的警示進入 `INSUFFICIENT_DATA` 狀態時通知您。如需詳細資訊，請參閱使用 CloudWatch 監控可擴展性資源的使用情況。

CloudWatch 也會追蹤 Application Auto Scaling 的 AWS API 用量指標。您可以使用這些指標來設定警示，即可在 API 呼叫量違反您定義的閾值時提醒您。如需詳細資訊，請參閱《Amazon CloudWatch 使用者指南》中的 AWS 用量指標。

Amazon EventBridge

Amazon EventBridge 為無伺服器事件匯流排服務，可讓您輕鬆將應用程式與來自各種來源的資料互相連線。EventBridge 會從您自己的應用程式、Software-as-a-Service(SaaS) 應用程式 AWS 和服務提供即時資料串流，並將該資料路由到 Lambda 等目標。這可以讓您監控在服務中發生的事件，並建置事件導向的架構。如需詳細資訊，請參閱使用 Amazon EventBridge 監控 Application Auto Scaling 事件。

AWS Health Dashboard

AWS Health Dashboard (PHD) 會顯示資訊，並提供由 AWS 資源運作狀態變更所叫用的通知。該資訊以兩種方式呈現：儀表板 (依類別顯示最近和近期事件) 和完整的事件日誌 (顯示過去 90 天內的所有事件)。如需詳細資訊，請參閱 入門 AWS Health Dashboard。

使用 CloudWatch 監控可擴展性資源的使用情況

使用 Amazon CloudWatch，您可以跨可擴展的資源獲得幾乎持續的應用程式可見性。CloudWatch 是 AWS 資源的監控服務。您可以使用 CloudWatch 收集和追蹤指標、設定警示及自動對 AWS 資源的變更做出反應。您也可以建立儀表板來監視需要的特定指標或指標集。

當您所互動的服務已與 Application Auto Scaling 整合時，這些服務會將下表所示的指標傳送至 CloudWatch。在 CloudWatch 中，指標會先依服務命名空間分組，再依各命名空間內不同的維度組合來分組。這些指標可協助您監控資源用量，以及規劃應用程式的容量。如果應用程式工作負載不固定，這表示您應該考慮使用自動擴展。如需這些指標的詳細說明，請參閱下表中感興趣之指標的說明文件。

目錄

用於監控資源用量的 CloudWatch 指標

下表列出可用於支援監控資源用量的 CloudWatch 指標。該清單並不詳盡，但會提供不錯的起點。如果您在 CloudWatch 主控台看不到這些指標，請確定您已完成資源的設定。如需更多資訊，請參閱 Amazon CloudWatch 使用者指南。

可擴展性資源	命名空間	CloudWatch 指標	文件的連結
AppStream 2.0			
機群	AWS/AppStream	名稱：AvailableCapacity	AppStream 2.0 指標

可擴展性資源	命名空間	CloudWatch 指標	文件的連結
		維度：機群	
機群	AWS/AppStream	名稱：CapacityUtilization 維度：機群	AppStream 2.0 指標
Aurora			
複本	AWS/RDS	名稱：CPUUtilization 維度：DBClusterIdentifier，Role (READER)	Aurora 叢集層級指標
複本	AWS/RDS	名稱：DatabaseConnections 維度：DBClusterIdentifier，Role (READER)	Aurora 叢集層級指標
Amazon Comprehend			

可擴展性資源	命名空間	CloudWatch 指標	文件的連結
文件分類端點	AWS/Comprehend	名稱：InferenceUtilization 維度：EndpointArn	Amazon Comprehend 端點指標
實體辨識器端點	AWS/Comprehend	名稱：InferenceUtilization 維度：EndpointArn	Amazon Comprehend 端點指標
DynamoDB			
資料表及全域次要索引	AWS/DynamoDB	名稱：ProvisionedReadCapacityUnits 維度：TableName、GlobalSecondaryIndexName	DynamoDB 指標

可擴展性資源	命名空間	CloudWatch 指標	文件的連結
資料表及全域次要索引	AWS/DynamoDB	名稱：ProvisionedWriteCapacityUnits 維度：TableName、GlobalSecondaryIndexName	DynamoDB 指標
資料表及全域次要索引	AWS/DynamoDB	名稱：ConsumedReadCapacityUnits 維度：TableName、GlobalSecondaryIndexName	DynamoDB 指標

可擴展性資源	命名空間	CloudWatch 指標	文件的連結
資料表及全域次要索引	AWS/DynamoDB	名稱：ConsumedWriteCapacityUnits 維度：TableName、GlobalSecondaryIndexName	DynamoDB 指標
Amazon ECS			
服務	AWS/ECS	名稱：CPUUtilization 維度：ClusterName、ServiceName	Amazon ECS 指標
服務	AWS/ECS	名稱：MemoryUtilization 維度：ClusterName、ServiceName	Amazon ECS 指標

可擴展性資源	命名空間	CloudWatch 指標	文件的連結
服務	AWS/ApplicationELB	名稱：RequestCountPerTarget 維度：TargetGroup	Application Load Balancer 指標
ElastiCache			
叢集 (複寫群組)	AWS/ElastiCache	名稱：DatabaseMemoryUsageCountedForEvictPercentage 維度：ReplicationGroupId	ElastiCache Valkey 和 Redis OSS 指標

可擴展性資源	命名空間	CloudWatch 指標	文件的連結
叢集 (複寫群組)	AWS/ElastiCache	名稱：DatabaseCapacityUsageCountedForEvictPercentage 維度：ReplicationGroupId	ElastiCache Valkey 和 Redis OSS 指標
叢集 (複寫群組)	AWS/ElastiCache	名稱：EngineCPUUtilization 維度：ReplicationGroupId, Role (Primary)	ElastiCache Valkey 和 Redis OSS 指標
叢集 (複寫群組)	AWS/ElastiCache	名稱：EngineCPUUtilization 維度：ReplicationGroupId, Role (Replica)	ElastiCache Valkey 和 Redis OSS 指標

可擴展性資源	命名空間	CloudWatch 指標	文件的連結
叢集（快取）	AWS/ElastiCache	名稱：EngineCPUUtilization 維度：CacheClusterId、節點	ElastiCache Memcached 指標
叢集（快取）	AWS/ElastiCache	名稱：DatabaseCapacityMemoryUsagePercentage 維度：CacheClusterId	ElastiCache Memcached 指標
Amazon EMR			
叢集	AWS/ElasticMapReduce	名稱：YARNMemoryAvailablePercentage 維度：ClusterId	Amazon EMR 指標
Amazon Keyspaces			

可擴展性資源	命名空間	CloudWatch 指標	文件的連結
資料表	AWS/Cassandra	名稱：ProvisionedReadCapacityUnits 維度：Keyspace，TableName	Amazon Keyspaces 指標
資料表	AWS/Cassandra	名稱：ProvisionedWriteCapacityUnits 維度：Keyspace，TableName	Amazon Keyspaces 指標
資料表	AWS/Cassandra	名稱：ConsumedReadCapacityUnits 維度：Keyspace，TableName	Amazon Keyspaces 指標

可擴展性資源	命名空間	CloudWatch 指標	文件的連結
資料表	AWS/Cassandra	名稱：ConsumedWriteCapacityUnits 維度：Keyspace，TableName	Amazon Keyspaces 指標
Lambda			
佈建並行	AWS/Lambda	名稱：ProvisionedConcurrencyUtilization 維度：FunctionName，Resource	Lambda 函數指標
Amazon MSK			
代理程式儲存	AWS/Kafka	名稱：KafkaDataLogsDiskUsed 維度：ClusterName	Amazon MSK 指標

可擴展性資源	命名空間	CloudWatch 指標	文件的連結
代理程式儲存	AWS/Kafka	名稱：KafkaDataLogsDiskUsed 維度：Cluster Name，Broker ID	Amazon MSK 指標
Neptune			
叢集	AWS/Neptune	名稱：CPUUtilization 維度：DBClusterIdentifier，Role (READER)	Neptune 指標
SageMaker AI			
端點變體	AWS/SageMaker	名稱：InvocationsPerInstance 維度：EndpointName，VariantName	呼叫指標

可擴展性資源	命名空間	CloudWatch 指標	文件的連結
推論元件	AWS/SageMaker	名稱：調用精子複製 維度：推論元件名稱	呼叫指標
無伺服器端點的佈建並行	AWS/SageMaker	名稱：ServerlessProvisionedConcurrencyUtilization 維度：EndpointName，VariantName	無伺服器端點指標
Spot 機群 (Amazon EC2)			
Spot Fleets	AWS/EC2Spot	名稱：CPUUtilization 維度：FleetRequestId	Spot 機群指標

可擴展性資源	命名空間	CloudWatch 指標	文件的連結
Spot Fleets	AWS/EC2Spot	名稱：NetworkIn 維度：FleetRequestId	Spot 機群指標
Spot Fleets	AWS/EC2Spot	名稱：NetworkOut 維度：FleetRequestId	Spot 機群指標
Spot Fleets	AWS/ApplicationELB	名稱：RequestCountPerTarget 維度：TargetGroup	Application Load Balancer 指標

目標追蹤擴展政策的預先定義指標

下表列出 Application Auto Scaling API 參考中預先定義的指標類型及其對應的 CloudWatch 指標名稱。每個預先定義的指標都代表基礎 CloudWatch 指標值的彙總。除非另有說明，否則結果是一分鐘期間內的平均資源用量，以百分比為基礎。預先定義的指標只能在設定目標追蹤擴展政策的內容中使用。

如需這些指標的詳細資訊，請參閱 用於監控資源用量的 CloudWatch 指標 中的資料表提供的所使用服務文件。

預先定義的指標類型	CloudWatch 指標名稱
AppStream 2.0	
`AppStreamAverageCapacityUtilization`	CapacityUtilization
Aurora	
`RDSReaderAverageCPUUtilization`	CPUUtilization
`RDSReaderAverageDatabaseConnections`	DatabaseConnections[1]
Amazon Comprehend	
`ComprehendInferenceUtilization`	InferenceUtilization
DynamoDB	
`DynamoDBReadCapacityUtilization`	ProvisionedReadCapacityUnits、ConsumedReadCapacityUnits[2]
`DynamoDBWriteCapacityUtilization`	ProvisionedWriteCapacityUnits、ConsumedWriteCapacityUnits[2]
Amazon ECS	
`ECSServiceAverageCPUUtilization`	CPUUtilization
`ECSServiceAverageMemoryUtilization`	MemoryUtilization
`ALBRequestCountPerTarget`	RequestCountPerTarget[1]
ElastiCache	
`ElastiCacheDatabaseMemoryUsageCountedForEvictPercentage`	DatabaseMemoryUsageCountedForEvictPercentage

預先定義的指標類型	CloudWatch 指標名稱
ElastiCacheDatabaseCapacityUsageCountedForEvictPercentage	DatabaseCapacityUsageCountedForEvictPercentage
ElastiCachePrimaryEngineCPUUtilization	EngineCPUUtilization
ElastiCacheReplicaEngineCPUUtilization	EngineCPUUtilization
ElastiCacheEngineCPUUtilization	EngineCPUUtilization
ElastiCacheDatabaseMemoryUsagePercentage	DatabaseMemoryUsagePercentage
Amazon Keyspaces	
CassandraReadCapacityUtilization	ProvisionedReadCapacityUnits、ConsumedReadCapacityUnits[2]
CassandraWriteCapacityUtilization	ProvisionedWriteCapacityUnits、ConsumedWriteCapacityUnits[2]
Lambda	
LambdaProvisionedConcurrencyUtilization	ProvisionedConcurrencyUtilization
Amazon MSK	
KafkaBrokerStorageUtilization	KafkaDataLogsDiskUsed
Neptune	
NeptuneReaderAverageCPUUtilization	CPUUtilization
SageMaker AI	

預先定義的指標類型	CloudWatch 指標名稱
SageMakerVariantInvocationsPerInstance	InvocationsPerInstance[1]
SageMakerInferenceComponentInvocationsPerCopy	調用員複製 [1]
SageMakerVariantProvisionedConcurrencyUtilization	ServerlessProvisionedConcurrencyUtilization
SageMakerInferenceComponentConcurrentRequestsPerCopyHighResolution	ConcurrentRequestsPerCopy
SageMakerVariantConcurrentRequestsPerModelHighResolution	ConcurrentRequestsPerModel
Spot 機群	
EC2SpotFleetRequestAverageCPUUtilization	CPUUtilization[3]
EC2SpotFleetRequestAverageNetworkIn[3]	NetworkIn[1] [3]
EC2SpotFleetRequestAverageNetworkOut[3]	NetworkOut[1] [3]
ALBRequestCountPerTarget	RequestCountPerTarget[1]

[1] 指標是以計數而非百分比為基礎。

[2] 對於 DynamoDB 和 Amazon Keyspaces，預先定義的指標是兩個 CloudWatch 指標的彙總，以支援根據佈建的輸送量消耗進行擴展。

[3] 為了獲得最佳的擴展效能，應該使用 Amazon EC2 詳細監控。

使用 記錄 Application Auto Scaling API 呼叫 AWS CloudTrail

Application Auto Scaling 已與 整合AWS CloudTrail，這項服務可提供由使用者、角色或 所採取動作的記錄 AWS 服務。CloudTrail 會將 Application Auto Scaling 的 API 呼叫擷取為事件。擷取的呼叫包括來自 的 AWS Management Console 呼叫，以及對 Application Auto Scaling API 操作的程式碼呼叫。您可以使用 CloudTrail 所收集的資訊，判斷對 Application Auto Scaling 提出的請求、提出請求的 IP 地址、提出請求的時間，以及其他詳細資訊。

每一筆事件或日誌專案都會包含產生請求者的資訊。身分資訊可協助您判斷下列事項：

- 該請求是使用根使用者還是使用者憑證提出。
- 請求是否代表 IAM Identity Center 使用者提出。
- 提出該請求時，是否使用了特定角色或聯合身分使用者的暫時安全憑證。
- 該請求是否由另一項 AWS 服務服務提出。

當您建立帳戶 AWS 帳戶 時CloudTrail 會在 中處於作用中狀態，而且您會自動存取 CloudTrail 事件歷史記錄。CloudTrail 事件歷史記錄為 AWS 區域中過去 90 天記錄的管理事件，提供可檢視、可搜尋、可下載且不可變的記錄。如需詳細資訊，請參閱「AWS CloudTrail 使用者指南」中的使用 CloudTrail 事件歷史記錄。檢視事件歷史記錄不會產生 CloudTrail 費用。

如需 AWS 帳戶 過去 90 天內持續記錄的事件，請建立線索。

CloudTrail 追蹤

線索能讓 CloudTrail 將日誌檔案交付至 Amazon S3 儲存貯體。使用 建立的所有線索 AWS Management Console 都是多區域。您可以使用 AWS CLI建立單一或多區域追蹤。建議您建立多區域線索，因為您擷取 AWS 區域 帳戶中所有 的活動。如果您建立單一區域追蹤，您只能檢視追蹤 AWS 區域中記錄的事件。如需追蹤的詳細資訊，請參閱《AWS CloudTrail 使用者指南》中的為您的 AWS 帳戶建立追蹤和為組織建立追蹤。

您可以透過建立追蹤，免費將持續管理事件的一個複本從 CloudTrail 傳遞至您的 Amazon S3 儲存貯體，但這樣做會產生 Amazon S3 儲存費用。如需 CloudTrail 定價的詳細資訊，請參閱 AWS CloudTrail 定價。如需 Amazon S3 定價的相關資訊，請參閱 Amazon S3 定價。

CloudTrail 中的 Application Auto Scaling 管理事件

管理事件提供在 資源上執行的管理操作的相關資訊 AWS 帳戶。這些也稱為控制平面操作。根據預設，CloudTrail 記錄管理事件。

Application Auto Scaling 會將所有 Application Auto Scaling 控制平面操作記錄為管理事件。如需 Application Auto Scaling 記錄到 CloudTrail 的 Application Auto Scaling 控制平面操作清單，請參閱 Application Auto Scaling API 參考。

Application Auto Scaling 事件範例

一個事件代表任何來源提出的單一請求，並包含請求 API 操作的相關資訊、操作的日期和時間、請求參數等。CloudTrail 日誌檔案不是公有 API 呼叫的已排序堆疊追蹤，因此事件不會以任何特定順序顯示。

以下範例顯示的 CloudTrail 事件會示範 `DescribeScalableTargets` 操作。

```
{
    "eventVersion": "1.05",
    "userIdentity": {
        "type": "Root",
        "principalId": "123456789012",
        "arn": "arn:aws:iam::123456789012:root",
        "accountId": "123456789012",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2018-08-21T17:05:42Z"
            }
        }
    },
    "eventTime": "2018-08-16T23:20:32Z",
    "eventSource": "autoscaling.amazonaws.com",
    "eventName": "DescribeScalableTargets",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "72.21.196.68",
    "userAgent": "EC2 Spot Console",
    "requestParameters": {
        "serviceNamespace": "ec2",
        "scalableDimension": "ec2:spot-fleet-request:TargetCapacity",
        "resourceIds": [
            "spot-fleet-request/sfr-05ceaf79-3ba2-405d-e87b-612857f1357a"
        ]
    },
    "responseElements": null,
    "additionalEventData": {
        "service": "application-autoscaling"
```

```
    },
    "requestID": "0737e2ea-fb2d-11e3-bfd8-99133058e7bb",
    "eventID": "3fcfb182-98f8-4744-bd45-b38835ab61cb",
    "eventType": "AwsApiCall",
    "recipientAccountId": "123456789012"
}
```

如需有關 CloudTrail 記錄內容的資訊，請參閱《AWS CloudTrail 使用者指南》中的 CloudTrail record contents。

CloudWatch 上的 Application Auto Scaling RemoveAction 呼叫

當 Application Auto Scaling 指示 CloudWatch 從警示中移除自動擴展動作時，您的 AWS CloudTrail 日誌可能會顯示 Application Auto Scaling 呼叫 CloudWatch `RemoveAction` API。如果您取消註冊可擴展的目標、刪除擴展政策，或警示調用不存在的擴展政策，就可能會發生這種情況。

使用 Amazon EventBridge 監控 Application Auto Scaling 事件

Amazon EventBridge (之前稱為 CloudWatch Events)，可幫助您監控 Application Auto Scaling 的特定事件，並啟動使用其他 AWS 服務的目標動作。來自 的事件 AWS 服務 會以近乎即時的方式交付至 EventBridge。

使用 EventBridge，您可以建立符合傳入事件的規則，並將其路由到目標以進行處理。

如需詳細資訊，請參閱 《Amazon EventBridge 使用者指南》中的 Amazon EventBridge 入門。

Application Auto Scaling 事件

以下範例顯示 Application Auto Scaling 的事件。盡可能產生事件。

目前只有明確要擴展到最大的事件和透過 CloudTrail 進行的 API 呼叫適用於 Application Auto Scaling。

事件類型

- 狀態變更的事件：擴展至最大值
- 透過 CloudTrail 進行 API 呼叫的事件

狀態變更的事件：擴展至最大值

下列範例事件顯示 Application Auto Scaling 將可擴展目標的容量增加 (橫向擴展) 到其大小上限。如果需求再次增加，Application Auto Scaling 無法將目標擴展到更大的大小，因為其已經擴展到大小上限。

在 `detail` 物件中，`resourceId`、`serviceNamespace` 和 `scalableDimension` 屬性的值標識可擴展的目標。`newDesiredCapacity` 和 `oldDesiredCapacity` 屬性的值是指橫向擴展事件之後的新容量以及橫向擴展事件之前的原始容量。`maxCapacity` 是可擴展目標的大小上限。

```
{
  "version": "0",
  "id": "11112222-3333-4444-5555-666677778888",
  "detail-type": "Application Auto Scaling Scaling Activity State Change",
  "source": "aws.application-autoscaling",
  "account": "123456789012",
  "time": "2019-06-12T10:23:40Z",
  "region": "us-west-2",
  "resources": [],
  "detail": {
    "startTime": "2022-06-12T10:20:43Z",
    "endTime": "2022-06-12T10:23:40Z",
    "newDesiredCapacity": 8,
    "oldDesiredCapacity": 5,
    "minCapacity": 2,
    "maxCapacity": 8,
    "resourceId": "table/my-table",
    "scalableDimension": "dynamodb:table:WriteCapacityUnits",
    "serviceNamespace": "dynamodb",
    "statusCode": "Successful",
    "scaledToMax": true,
    "direction": "scale-out"
}
```

若要建立擷取所有可擴展目標的全部 `scaledToMax` 狀態變更事件的規則，請使用下列範例事件模式。

```
{
  "source": [
    "aws.application-autoscaling"
  ],
  "detail-type": [
```

```
    "Application Auto Scaling Scaling Activity State Change"
  ],
  "detail": {
    "scaledToMax": [
      true
    ]
  }
}
```

透過 CloudTrail 進行 API 呼叫的事件

線索是一種組態， AWS CloudTrail 使用 將事件做為日誌檔案交付至 Amazon S3 儲存貯體。CloudTrail 日誌檔案包含日誌項目。一個事件代表一個日誌項目，它包含所要求動作、動作日期和時間以及要求參數等資訊。如需了解如何開始使用 CloudTrail，請參閱 AWS CloudTrail 使用者指南中的建立追蹤。

透過 CloudTrail 交付的事件，其 `detail-type` 的值都會是 `AWS API Call via CloudTrail`。

下列範例事件代表 CloudTrail 日誌檔案項目，其中顯示呼叫 Application Auto Scaling RegisterScalableTarget 動作的主控台使用者。

```
{
  "version": "0",
  "id": "99998888-7777-6666-5555-444433332222",
  "detail-type": "AWS API Call via CloudTrail",
  "source": "aws.autoscaling",
  "account": "123456789012",
  "time": "2022-07-13T16:50:15Z",
  "region": "us-west-2",
  "resources": [],
  "detail": {
    "eventVersion": "1.08",
    "userIdentity": {
      "type": "IAMUser",
      "principalId": "123456789012",
      "arn": "arn:aws:iam::123456789012:user/Bob",
      "accountId": "123456789012",
      "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
      "sessionContext": {
        "sessionIssuer": {
          "type": "Role",
          "principalId": "123456789012",
          "arn": "arn:aws:iam::123456789012:role/Admin",
```

```
          "accountId": "123456789012",
          "userName": "Admin"
        },
        "webIdFederationData": {},
        "attributes": {
          "creationDate": "2022-07-13T15:17:08Z",
          "mfaAuthenticated": "false"
        }
      }
    },
    "eventTime": "2022-07-13T16:50:15Z",
    "eventSource": "autoscaling.amazonaws.com",
    "eventName": "RegisterScalableTarget",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "AWS Internal",
    "userAgent": "EC2 Spot Console",
    "requestParameters": {
      "resourceId": "spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE",
      "serviceNamespace": "ec2",
      "scalableDimension": "ec2:spot-fleet-request:TargetCapacity",
      "minCapacity": 2,
      "maxCapacity": 10
    },
    "responseElements": null,
    "additionalEventData": {
      "service": "application-autoscaling"
    },
    "requestID": "e9caf887-8d88-11e5-a331-3332aa445952",
    "eventID": "49d14f36-6450-44a5-a501-b0fdcdfaeb98",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "eventCategory": "Management",
    "sessionCredentialFromConsole": "true"
  }
}
```

若要根據所有可擴展目標的 DeleteScalingPolicy 和 DeregisterScalableTarget API 呼叫建立規則，請使用以下範例事件模式：

```
{
  "source": [
```

```
    "aws.autoscaling"
  ],
  "detail-type": [
    "AWS API Call via CloudTrail"
  ],
  "detail": {
    "eventSource": [
      "autoscaling.amazonaws.com"
    ],
    "eventName": [
       "DeleteScalingPolicy",
       "DeregisterScalableTarget"
    ],
    "additionalEventData": {
      "service": [
        "application-autoscaling"
      ]
    }
  }
}
```

如需使用 CloudTrail 的詳細資訊，請參閱「使用 記錄 Application Auto Scaling API 呼叫 AWS CloudTrail」。

搭配 AWS SDK 使用此服務

AWS 軟體開發套件 (SDKs) 適用於許多熱門的程式設計語言。每個 SDK 都提供 API、程式碼範例和說明文件，讓開發人員能夠更輕鬆地以偏好的語言建置應用程式。

SDK 文件	代碼範例
適用於 C++ 的 AWS SDK	適用於 C++ 的 AWS SDK 程式碼範例
AWS CLI	AWS CLI 程式碼範例
適用於 Go 的 AWS SDK	適用於 Go 的 AWS SDK 程式碼範例
適用於 Java 的 AWS SDK	適用於 Java 的 AWS SDK 程式碼範例
適用於 JavaScript 的 AWS SDK	適用於 JavaScript 的 AWS SDK 程式碼範例
適用於 Kotlin 的 AWS SDK	適用於 Kotlin 的 AWS SDK 程式碼範例
適用於 .NET 的 AWS SDK	適用於 .NET 的 AWS SDK 程式碼範例
適用於 PHP 的 AWS SDK	適用於 PHP 的 AWS SDK 程式碼範例
AWS Tools for PowerShell	AWS Tools for PowerShell 程式碼範例
適用於 Python (Boto3) 的 AWS SDK	適用於 Python (Boto3) 的 AWS SDK 程式碼範例
適用於 Ruby 的 AWS SDK	適用於 Ruby 的 AWS SDK 程式碼範例
適用於 Rust 的 AWS SDK	適用於 Rust 的 AWS SDK 程式碼範例
適用於 SAP ABAP 的 AWS SDK	適用於 SAP ABAP 的 AWS SDK 程式碼範例
適用於 Swift 的 AWS SDK	適用於 Swift 的 AWS SDK 程式碼範例

可用性範例

找不到所需的內容嗎？ 請使用本頁面底部的提供意見回饋連結申請程式碼範例。

Application Auto Scaling AWS SDKs程式碼範例

下列程式碼範例示範如何使用 Application Auto Scaling 搭配 AWS 軟體開發套件 (SDK)。

Actions 是大型程式的程式碼摘錄，必須在內容中執行。雖然動作會告訴您如何呼叫個別服務函數，但您可以在其相關情境中查看內容中的動作。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

程式碼範例

- Application Auto Scaling AWS SDKs的基本範例
  - 使用 AWS SDKs進行 Application Auto Scaling 的動作
    - DeleteScalingPolicy 搭配 AWS SDK 或 CLI 使用
    - 搭配使用 DeleteScheduledAction 與 CLI
    - 搭配使用 DeregisterScalableTarget 與 CLI
    - 搭配使用 DescribeScalableTargets 與 CLI
    - 搭配使用 DescribeScalingActivities 與 CLI
    - DescribeScalingPolicies 搭配 AWS SDK 或 CLI 使用
    - 搭配使用 DescribeScheduledActions 與 CLI
    - 搭配使用 PutScalingPolicy 與 CLI
    - 搭配使用 PutScheduledAction 與 CLI
    - RegisterScalableTarget 搭配 AWS SDK 或 CLI 使用

Application Auto Scaling AWS SDKs的基本範例

下列程式碼範例示範如何使用 Application Auto Scaling 與 AWS SDKs的基本概念。

範例

- 使用 AWS SDKs進行 Application Auto Scaling 的動作
  - DeleteScalingPolicy 搭配 AWS SDK 或 CLI 使用
  - 搭配使用 DeleteScheduledAction 與 CLI
  - 搭配使用 DeregisterScalableTarget 與 CLI
  - 搭配使用 DescribeScalableTargets 與 CLI

- 搭配使用 DescribeScalingActivities 與 CLI
- DescribeScalingPolicies 搭配 AWS SDK 或 CLI 使用
- 搭配使用 DescribeScheduledActions 與 CLI
- 搭配使用 PutScalingPolicy 與 CLI
- 搭配使用 PutScheduledAction 與 CLI
- RegisterScalableTarget 搭配 AWS SDK 或 CLI 使用

使用 AWS SDKs進行 Application Auto Scaling 的動作

下列程式碼範例示範如何使用 AWS SDKs 執行個別 Application Auto Scaling 動作。每個範例均包含 GitHub 的連結，您可以在連結中找到設定和執行程式碼的相關說明。

下列範例僅包含最常使用的動作。如需完整清單，請參閱 Application Auto Scaling API 參考。

範例

- DeleteScalingPolicy 搭配 AWS SDK 或 CLI 使用
- 搭配使用 DeleteScheduledAction 與 CLI
- 搭配使用 DeregisterScalableTarget 與 CLI
- 搭配使用 DescribeScalableTargets 與 CLI
- 搭配使用 DescribeScalingActivities 與 CLI
- DescribeScalingPolicies 搭配 AWS SDK 或 CLI 使用
- 搭配使用 DescribeScheduledActions 與 CLI
- 搭配使用 PutScalingPolicy 與 CLI
- 搭配使用 PutScheduledAction 與 CLI
- RegisterScalableTarget 搭配 AWS SDK 或 CLI 使用

`DeleteScalingPolicy` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `DeleteScalingPolicy`。

CLI

AWS CLI

刪除擴展政策

此範例會刪除在預設叢集中執行的 Amazon ECS 服務 Web 應用程式擴展政策。

命令：

```
aws application-autoscaling delete-scaling-policy --policy-name web-app-cpu-lt-25
 --scalable-dimension ecs:service:DesiredCount --resource-id service/default/web-
app --service-namespace ecs
```

- 如需 API 詳細資訊，請參閱《 AWS CLI 命令參考》中的 DeleteScalingPolicy。

Java

SDK for Java 2.x

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import software.amazon.awssdk.regions.Region;
import
 software.amazon.awssdk.services.applicationautoscaling.ApplicationAutoScalingClient;
import
 software.amazon.awssdk.services.applicationautoscaling.model.ApplicationAutoScalingExcep
import
 software.amazon.awssdk.services.applicationautoscaling.model.DeleteScalingPolicyRequest;
import
 software.amazon.awssdk.services.applicationautoscaling.model.DeregisterScalableTargetReq
import
 software.amazon.awssdk.services.applicationautoscaling.model.DescribeScalableTargetsRequ
import
 software.amazon.awssdk.services.applicationautoscaling.model.DescribeScalableTargetsResp
import
 software.amazon.awssdk.services.applicationautoscaling.model.DescribeScalingPoliciesRequ
import
 software.amazon.awssdk.services.applicationautoscaling.model.DescribeScalingPoliciesResp
import
 software.amazon.awssdk.services.applicationautoscaling.model.ScalableDimension;
import
 software.amazon.awssdk.services.applicationautoscaling.model.ServiceNamespace;
```

```
/**
 * Before running this Java V2 code example, set up your development environment,
 including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */

public class DisableDynamoDBAutoscaling {
    public static void main(String[] args) {
        final String usage = """

            Usage:
               <tableId> <policyName>\s

            Where:
               tableId - The table Id value (for example, table/Music).\s
               policyName - The name of the policy (for example, $Music5-scaling-
policy).

            """;
        if (args.length != 2) {
            System.out.println(usage);
            System.exit(1);
        }

        ApplicationAutoScalingClient appAutoScalingClient =
 ApplicationAutoScalingClient.builder()
            .region(Region.US_EAST_1)
            .build();

        ServiceNamespace ns = ServiceNamespace.DYNAMODB;
        ScalableDimension tableWCUs =
 ScalableDimension.DYNAMODB_TABLE_WRITE_CAPACITY_UNITS;
        String tableId = args[0];
        String policyName = args[1];

        deletePolicy(appAutoScalingClient, policyName, tableWCUs, ns, tableId);
        verifyScalingPolicies(appAutoScalingClient, tableId, ns, tableWCUs);
        deregisterScalableTarget(appAutoScalingClient, tableId, ns, tableWCUs);
        verifyTarget(appAutoScalingClient, tableId, ns, tableWCUs);
```

```
   }

   public static void deletePolicy(ApplicationAutoScalingClient
appAutoScalingClient, String policyName, ScalableDimension tableWCUs,
ServiceNamespace ns, String tableId) {
       try {
           DeleteScalingPolicyRequest delSPRequest =
DeleteScalingPolicyRequest.builder()
               .policyName(policyName)
               .scalableDimension(tableWCUs)
               .serviceNamespace(ns)
               .resourceId(tableId)
               .build();

           appAutoScalingClient.deleteScalingPolicy(delSPRequest);
           System.out.println(policyName +" was deleted successfully.");

       } catch (ApplicationAutoScalingException e) {
           System.err.println(e.awsErrorDetails().errorMessage());
       }
   }

   // Verify that the scaling policy was deleted
   public static void verifyScalingPolicies(ApplicationAutoScalingClient
appAutoScalingClient, String tableId, ServiceNamespace ns, ScalableDimension
tableWCUs) {
       DescribeScalingPoliciesRequest dscRequest =
DescribeScalingPoliciesRequest.builder()
           .scalableDimension(tableWCUs)
           .serviceNamespace(ns)
           .resourceId(tableId)
           .build();

       DescribeScalingPoliciesResponse response =
appAutoScalingClient.describeScalingPolicies(dscRequest);
       System.out.println("DescribeScalableTargets result: ");
       System.out.println(response);
   }

   public static void deregisterScalableTarget(ApplicationAutoScalingClient
appAutoScalingClient, String tableId, ServiceNamespace ns, ScalableDimension
tableWCUs) {
       try {
```

```
            DeregisterScalableTargetRequest targetRequest =
 DeregisterScalableTargetRequest.builder()
                .scalableDimension(tableWCUs)
                .serviceNamespace(ns)
                .resourceId(tableId)
                .build();

            appAutoScalingClient.deregisterScalableTarget(targetRequest);
            System.out.println("The scalable target was deregistered.");

        } catch (ApplicationAutoScalingException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
        }
    }

    public static void verifyTarget(ApplicationAutoScalingClient
 appAutoScalingClient, String tableId, ServiceNamespace ns, ScalableDimension
 tableWCUs) {
        DescribeScalableTargetsRequest dscRequest =
 DescribeScalableTargetsRequest.builder()
            .scalableDimension(tableWCUs)
            .serviceNamespace(ns)
            .resourceIds(tableId)
            .build();

        DescribeScalableTargetsResponse response =
 appAutoScalingClient.describeScalableTargets(dscRequest);
        System.out.println("DescribeScalableTargets result: ");
        System.out.println(response);
    }
}
```

- 如需 API 詳細資訊，請參閱AWS SDK for Java 2.x 《 API 參考》中的 DeleteScalingPolicy。

PowerShell

PowerShell V4 的工具

範例 1：此 cmdlet 會刪除 Application Auto Scaling 可擴展目標的指定擴展政策。

```
Remove-AASScalingPolicy -ServiceNamespace AppStream -PolicyName "default-scale-
out" -ResourceId fleet/Test -ScalableDimension appstream:fleet:DesiredCapacity
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V4)》中的 DeleteScalingPolicy。

PowerShell V5 的工具

範例 1：此 cmdlet 會刪除 Application Auto Scaling 可擴展目標的指定擴展政策。

```
Remove-AASScalingPolicy -ServiceNamespace AppStream -PolicyName "default-scale-
out" -ResourceId fleet/Test -ScalableDimension appstream:fleet:DesiredCapacity
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet 參考 (V5) 中的 DeleteScalingPolicy。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **DeleteScheduledAction** 與 CLI

下列程式碼範例示範如何使用 DeleteScheduledAction。

CLI

AWS CLI

刪除排程動作

模糊delete-scheduled-action範例會從指定的 Amazon AppStream 2.0 機群刪除指定的排程動作：

```
aws application-autoscaling delete-scheduled-action \
    --service-namespace appstream \
    --scalable-dimension appstream:fleet:DesiredCapacity \
    --resource-id fleet/sample-fleet \
    --scheduled-action-name my-recurring-action
```

此命令不會產生輸出。

如需詳細資訊，請參閱《應用程式自動擴展使用者指南》中的排程擴展。

- 如需 API 詳細資訊，請參閱《 AWS CLI 命令參考》中的 DeleteScheduledAction。

PowerShell

PowerShell V4 的工具

範例 1：此 cmdlet 會刪除 Application Auto Scaling 可擴展目標的指定排程動作。

```
Remove-AASScheduledAction -ServiceNamespace AppStream -ScheduledActionName
 WeekDaysFleetScaling -ResourceId fleet/MyFleet -ScalableDimension
 appstream:fleet:DesiredCapacity
```

輸出：

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-AASScheduledAction (DeleteScheduledAction)" on
 target "WeekDaysFleetScaling".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"): Y
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V4)》中的 DeleteScheduledAction。

PowerShell V5 的工具

範例 1：此 cmdlet 會刪除 Application Auto Scaling 可擴展目標的指定排程動作。

```
Remove-AASScheduledAction -ServiceNamespace AppStream -ScheduledActionName
 WeekDaysFleetScaling -ResourceId fleet/MyFleet -ScalableDimension
 appstream:fleet:DesiredCapacity
```

輸出：

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-AASScheduledAction (DeleteScheduledAction)" on
 target "WeekDaysFleetScaling".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"): Y
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V5)》中的 DeleteScheduledAction。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **DeregisterScalableTarget** 與 CLI

下列程式碼範例示範如何使用 DeregisterScalableTarget。

CLI

AWS CLI

取消註冊可擴展的目標

此範例會取消註冊在預設叢集中執行之稱為 Web 應用程式之 Amazon ECS 服務的可擴展目標。

命令：

```
aws application-autoscaling deregister-scalable-target --service-namespace ecs --
scalable-dimension ecs:service:DesiredCount --resource-id service/default/web-app
```

此範例會取消註冊自訂資源的可擴展目標。custom-resource-id.txt 檔案包含可識別資源 ID 的字串，對於自訂資源，這是透過 Amazon API Gateway 端點存取自訂資源的路徑。

命令：

```
aws application-autoscaling deregister-scalable-target --
service-namespace custom-resource --scalable-dimension custom-
resource:ResourceType:Property --resource-id file://~/custom-resource-id.txt
```

custom-resource-id.txt 檔案的內容：

```
https://example.execute-api.us-west-2.amazonaws.com/prod/
scalableTargetDimensions/1-23456789
```

- 如需 API 詳細資訊，請參閱《 AWS CLI 命令參考》中的 DeregisterScalableTarget。

PowerShell

PowerShell V4 的工具

範例 1：此 cmdlet 會取消註冊 Application Auto Scaling 可擴展目標。取消註冊可擴展目標會刪除與其相關聯的擴展政策。

```
Remove-AASScalableTarget -ResourceId fleet/MyFleet -ScalableDimension
 appstream:fleet:DesiredCapacity -ServiceNamespace AppStream
```

輸出：

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-AASScalableTarget (DeregisterScalableTarget)" on
 target "fleet/MyFleet".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"): Y
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet 參考 (V4) 中的 DeregisterScalableTarget。

PowerShell V5 的工具

範例 1：此 cmdlet 會取消註冊 Application Auto Scaling 可擴展目標。取消註冊可擴展目標會刪除與其相關聯的擴展政策。

```
Remove-AASScalableTarget -ResourceId fleet/MyFleet -ScalableDimension
 appstream:fleet:DesiredCapacity -ServiceNamespace AppStream
```

輸出：

```
Confirm
Are you sure you want to perform this action?
Performing the operation "Remove-AASScalableTarget (DeregisterScalableTarget)" on
 target "fleet/MyFleet".
[Y] Yes  [A] Yes to All  [N] No  [L] No to All  [S] Suspend  [?] Help (default is
 "Y"): Y
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet 參考 (V5) 中的 DeregisterScalableTarget。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **DescribeScalableTargets** 與 CLI

下列程式碼範例示範如何使用 DescribeScalableTargets。

CLI

AWS CLI

描述可擴展的目標

下列describe-scalable-targets範例說明ecs服務命名空間的可擴展目標。

```
aws application-autoscaling describe-scalable-targets \
    --service-namespace ecs
```

輸出：

```
{
    "ScalableTargets": [
        {
            "ServiceNamespace": "ecs",
            "ScalableDimension": "ecs:service:DesiredCount",
            "ResourceId": "service/default/web-app",
            "MinCapacity": 1,
            "MaxCapacity": 10,
            "RoleARN": "arn:aws:iam::123456789012:role/
aws-service-role/ecs.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_ECSService",
            "CreationTime": 1462558906.199,
            "SuspendedState": {
                "DynamicScalingOutSuspended": false,
                "ScheduledScalingSuspended": false,
                "DynamicScalingInSuspended": false
            },
            "ScalableTargetARN": "arn:aws:application-autoscaling:us-
west-2:123456789012:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
        }
    ]
}
```

如需詳細資訊，請參閱《AWS Application Auto Scaling 使用者指南》中的可與 Application Auto Scaling 搭配使用的 服務。 Auto Scaling

- 如需 API 詳細資訊，請參閱《 AWS CLI 命令參考》中的 DescribeScalableTargets。

PowerShell

PowerShell V4 的工具

範例 1：此範例將提供指定命名空間中 Application Autoscaling 可擴展目標的相關資訊。

```
Get-AASScalableTarget -ServiceNamespace "AppStream"
```

輸出：

```
CreationTime      : 11/7/2019 2:30:03 AM
MaxCapacity       : 5
MinCapacity       : 1
ResourceId        : fleet/Test
RoleARN           : arn:aws:iam::012345678912:role/aws-
service-role/appstream.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_AppStreamFleet
ScalableDimension : appstream:fleet:DesiredCapacity
ServiceNamespace  : appstream
SuspendedState    : Amazon.ApplicationAutoScaling.Model.SuspendedState
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V4) 中的 DescribeScalableTargets。

PowerShell V5 的工具

範例 1：此範例將提供指定命名空間中 Application Autoscaling 可擴展目標的相關資訊。

```
Get-AASScalableTarget -ServiceNamespace "AppStream"
```

輸出：

```
CreationTime      : 11/7/2019 2:30:03 AM
MaxCapacity       : 5
MinCapacity       : 1
ResourceId        : fleet/Test
```

```
RoleARN           : arn:aws:iam::012345678912:role/aws-
service-role/appstream.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_AppStreamFleet
ScalableDimension : appstream:fleet:DesiredCapacity
ServiceNamespace  : appstream
SuspendedState    : Amazon.ApplicationAutoScaling.Model.SuspendedState
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet 參考 (V5) 中的 DescribeScalableTargets。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **DescribeScalingActivities** 與 CLI

下列程式碼範例示範如何使用 DescribeScalingActivities。

CLI

AWS CLI

範例 1：描述指定 Amazon ECS 服務的擴展活動

下列describe-scaling-activities範例說明叢集中執行web-app之名為 的 Amazon ECS 服務擴展活動default。輸出會顯示由擴展政策啟動的擴展活動。

```
aws application-autoscaling describe-scaling-activities \
    --service-namespace ecs \
    --resource-id service/default/web-app
```

輸出：

```
{
    "ScalingActivities": [
        {
            "ScalableDimension": "ecs:service:DesiredCount",
            "Description": "Setting desired count to 1.",
            "ResourceId": "service/default/web-app",
            "ActivityId": "e6c5f7d1-dbbb-4a3f-89b2-51f33e766399",
            "StartTime": 1462575838.171,
            "ServiceNamespace": "ecs",
```

```
            "EndTime": 1462575872.111,
            "Cause": "monitor alarm web-app-cpu-lt-25 in state ALARM triggered
 policy web-app-cpu-lt-25",
            "StatusMessage": "Successfully set desired count to 1. Change
 successfully fulfilled by ecs.",
            "StatusCode": "Successful"
        }
    ]
}
```

如需詳細資訊，請參閱《Application Auto Scaling 使用者指南》中的 Application Auto Scaling 的擴展活動。

範例 2：描述指定 DynamoDB 資料表的擴展活動

下列`describe-scaling-activities`範例說明名為 之 DynamoDB 資料表的擴展活動`TestTable`。輸出會顯示由兩個不同的排程動作啟動的擴展活動。

```
aws application-autoscaling describe-scaling-activities \
    --service-namespace dynamodb \
    --resource-id table/TestTable
```

輸出：

```
{
    "ScalingActivities": [
        {
            "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
            "Description": "Setting write capacity units to 10.",
            "ResourceId": "table/my-table",
            "ActivityId": "4d1308c0-bbcf-4514-a673-b0220ae38547",
            "StartTime": 1561574415.086,
            "ServiceNamespace": "dynamodb",
            "EndTime": 1561574449.51,
            "Cause": "maximum capacity was set to 10",
            "StatusMessage": "Successfully set write capacity units to 10. Change
 successfully fulfilled by dynamodb.",
            "StatusCode": "Successful"
        },
        {
            "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
            "Description": "Setting min capacity to 5 and max capacity to 10",
```

```
            "ResourceId": "table/my-table",
            "ActivityId": "f2b7847b-721d-4e01-8ef0-0c8d3bacc1c7",
            "StartTime": 1561574414.644,
            "ServiceNamespace": "dynamodb",
            "Cause": "scheduled action name my-second-scheduled-action was
 triggered",
            "StatusMessage": "Successfully set min capacity to 5 and max capacity
 to 10",
            "StatusCode": "Successful"
        },
        {
            "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
            "Description": "Setting write capacity units to 15.",
            "ResourceId": "table/my-table",
            "ActivityId": "d8ea4de6-9eaa-499f-b466-2cc5e681ba8b",
            "StartTime": 1561574108.904,
            "ServiceNamespace": "dynamodb",
            "EndTime": 1561574140.255,
            "Cause": "minimum capacity was set to 15",
            "StatusMessage": "Successfully set write capacity units to 15. Change
 successfully fulfilled by dynamodb.",
            "StatusCode": "Successful"
        },
        {
            "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
            "Description": "Setting min capacity to 15 and max capacity to 20",
            "ResourceId": "table/my-table",
            "ActivityId": "3250fd06-6940-4e8e-bb1f-d494db7554d2",
            "StartTime": 1561574108.512,
            "ServiceNamespace": "dynamodb",
            "Cause": "scheduled action name my-first-scheduled-action was
 triggered",
            "StatusMessage": "Successfully set min capacity to 15 and max
 capacity to 20",
            "StatusCode": "Successful"
        }
    ]
}
```

如需詳細資訊，請參閱《Application Auto Scaling 使用者指南》中的 Application Auto Scaling 的擴展活動。

- 如需 API 詳細資訊，請參閱《 AWS CLI 命令參考》中的 DescribeScalingActivities。

PowerShell

PowerShell V4 的工具

範例 1：提供過去六週內指定命名空間中擴展活動的描述性資訊。

```
Get-AASScalingActivity -ServiceNamespace AppStream
```

輸出：

```
ActivityId        : 2827409f-b639-4cdb-a957-8055d5d07434
Cause             : monitor alarm Appstream2-MyFleet-default-scale-in-Alarm in
 state ALARM triggered policy default-scale-in
Description       : Setting desired capacity to 2.
Details           :
EndTime           : 12/14/2019 11:32:49 AM
ResourceId        : fleet/MyFleet
ScalableDimension : appstream:fleet:DesiredCapacity
ServiceNamespace  : appstream
StartTime         : 12/14/2019 11:32:14 AM
StatusCode        : Successful
StatusMessage     : Successfully set desired capacity to 2. Change successfully
 fulfilled by appstream.
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V4) 中的 DescribeScalingActivities。

PowerShell V5 的工具

範例 1：提供過去六週內指定命名空間中擴展活動的描述性資訊。

```
Get-AASScalingActivity -ServiceNamespace AppStream
```

輸出：

```
ActivityId        : 2827409f-b639-4cdb-a957-8055d5d07434
Cause             : monitor alarm Appstream2-MyFleet-default-scale-in-Alarm in
 state ALARM triggered policy default-scale-in
Description       : Setting desired capacity to 2.
Details           :
EndTime           : 12/14/2019 11:32:49 AM
ResourceId        : fleet/MyFleet
ScalableDimension : appstream:fleet:DesiredCapacity
```

```
ServiceNamespace  : appstream
StartTime         : 12/14/2019 11:32:14 AM
StatusCode        : Successful
StatusMessage     : Successfully set desired capacity to 2. Change successfully
 fulfilled by appstream.
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V5) 中的 DescribeScalingActivities。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`DescribeScalingPolicies` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `DescribeScalingPolicies`。

CLI

AWS CLI

描述擴展政策

此範例命令說明 ecs 服務命名空間的擴展政策。

命令：

```
aws application-autoscaling describe-scaling-policies --service-namespace ecs
```

輸出：

```
{
    "ScalingPolicies": [
        {
            "PolicyName": "web-app-cpu-gt-75",
            "ScalableDimension": "ecs:service:DesiredCount",
            "ResourceId": "service/default/web-app",
            "CreationTime": 1462561899.23,
            "StepScalingPolicyConfiguration": {
                "Cooldown": 60,
                "StepAdjustments": [
                    {
                        "ScalingAdjustment": 200,
```

```
                        "MetricIntervalLowerBound": 0.0
                    }
                ],
                "AdjustmentType": "PercentChangeInCapacity"
            },
            "PolicyARN": "arn:aws:autoscaling:us-
west-2:012345678910:scalingPolicy:6d8972f3-efc8-437c-92d1-6270f29a66e7:resource/
ecs/service/default/web-app:policyName/web-app-cpu-gt-75",
            "PolicyType": "StepScaling",
            "Alarms": [
                {
                    "AlarmName": "web-app-cpu-gt-75",
                    "AlarmARN": "arn:aws:cloudwatch:us-
west-2:012345678910:alarm:web-app-cpu-gt-75"
                }
            ],
            "ServiceNamespace": "ecs"
        },
        {
            "PolicyName": "web-app-cpu-lt-25",
            "ScalableDimension": "ecs:service:DesiredCount",
            "ResourceId": "service/default/web-app",
            "CreationTime": 1462562575.099,
            "StepScalingPolicyConfiguration": {
                "Cooldown": 1,
                "StepAdjustments": [
                    {
                        "ScalingAdjustment": -50,
                        "MetricIntervalUpperBound": 0.0
                    }
                ],
                "AdjustmentType": "PercentChangeInCapacity"
            },
            "PolicyARN": "arn:aws:autoscaling:us-
west-2:012345678910:scalingPolicy:6d8972f3-efc8-437c-92d1-6270f29a66e7:resource/
ecs/service/default/web-app:policyName/web-app-cpu-lt-25",
            "PolicyType": "StepScaling",
            "Alarms": [
                {
                    "AlarmName": "web-app-cpu-lt-25",
                    "AlarmARN": "arn:aws:cloudwatch:us-
west-2:012345678910:alarm:web-app-cpu-lt-25"
                }
            ],
```

```
            "ServiceNamespace": "ecs"
        }
    ]
}
```

- 如需 API 詳細資訊，請參閱《 AWS CLI 命令參考》中的 DescribeScalingPolicies。

PowerShell

PowerShell V4 的工具

範例 1：此 cmdlet 說明指定服務命名空間的 Application Auto Scaling 擴展政策。

```
Get-AASScalingPolicy -ServiceNamespace AppStream
```

輸出：

```
Alarms                                    : {Appstream2-LabFleet-default-scale-
out-Alarm}
CreationTime                              : 9/3/2019 2:48:15 AM
PolicyARN                                 : arn:aws:autoscaling:us-
west-2:012345678912:scalingPolicy:5659b069-b5cd-4af1-9f7f-3e956d36233e:resource/
appstream/fleet/LabFleet:
                                            policyName/default-scale-out
PolicyName                                : default-scale-out
PolicyType                                : StepScaling
ResourceId                                : fleet/LabFleet
ScalableDimension                         : appstream:fleet:DesiredCapacity
ServiceNamespace                          : appstream
StepScalingPolicyConfiguration            :
 Amazon.ApplicationAutoScaling.Model.StepScalingPolicyConfiguration
TargetTrackingScalingPolicyConfiguration :

Alarms                                    : {Appstream2-LabFleet-default-scale-in-
Alarm}
CreationTime                              : 9/3/2019 2:48:15 AM
PolicyARN                                 : arn:aws:autoscaling:us-
west-2:012345678912:scalingPolicy:5659b069-b5cd-4af1-9f7f-3e956d36233e:resource/
appstream/fleet/LabFleet:
                                            policyName/default-scale-in
PolicyName                                : default-scale-in
PolicyType                                : StepScaling
```

```
ResourceId                                : fleet/LabFleet
ScalableDimension                         : appstream:fleet:DesiredCapacity
ServiceNamespace                          : appstream
StepScalingPolicyConfiguration            :
 Amazon.ApplicationAutoScaling.Model.StepScalingPolicyConfiguration
TargetTrackingScalingPolicyConfiguration :
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V4) 中的 DescribeScalingPolicies。

PowerShell V5 的工具

範例 1：此 cmdlet 說明指定服務命名空間的 Application Auto Scaling 擴展政策。

```
Get-AASScalingPolicy -ServiceNamespace AppStream
```

輸出：

```
Alarms                                    : {Appstream2-LabFleet-default-scale-
out-Alarm}
CreationTime                              : 9/3/2019 2:48:15 AM
PolicyARN                                 : arn:aws:autoscaling:us-
west-2:012345678912:scalingPolicy:5659b069-b5cd-4af1-9f7f-3e956d36233e:resource/
appstream/fleet/LabFleet:
                                            policyName/default-scale-out
PolicyName                                : default-scale-out
PolicyType                                : StepScaling
ResourceId                                : fleet/LabFleet
ScalableDimension                         : appstream:fleet:DesiredCapacity
ServiceNamespace                          : appstream
StepScalingPolicyConfiguration            :
 Amazon.ApplicationAutoScaling.Model.StepScalingPolicyConfiguration
TargetTrackingScalingPolicyConfiguration :

Alarms                                    : {Appstream2-LabFleet-default-scale-in-
Alarm}
CreationTime                              : 9/3/2019 2:48:15 AM
PolicyARN                                 : arn:aws:autoscaling:us-
west-2:012345678912:scalingPolicy:5659b069-b5cd-4af1-9f7f-3e956d36233e:resource/
appstream/fleet/LabFleet:
                                            policyName/default-scale-in
PolicyName                                : default-scale-in
PolicyType                                : StepScaling
ResourceId                                : fleet/LabFleet
```

```
ScalableDimension                       : appstream:fleet:DesiredCapacity
ServiceNamespace                        : appstream
StepScalingPolicyConfiguration          :
 Amazon.ApplicationAutoScaling.Model.StepScalingPolicyConfiguration
TargetTrackingScalingPolicyConfiguration :
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet 參考 (V5) 中的 DescribeScalingPolicies。

Rust

SDK for Rust

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
async fn show_policies(client: &Client) -> Result<(), Error> {
    let response = client
        .describe_scaling_policies()
        .service_namespace(ServiceNamespace::Ec2)
        .send()
        .await?;
    println!("Auto Scaling Policies:");
    for policy in response.scaling_policies() {
        println!("{:?}\n", policy);
    }
    println!("Next token: {:?}", response.next_token());

    Ok(())
}
```

- 如需 API 詳細資訊，請參閱《適用於 AWS Rust 的 SDK API 參考》中的 DescribeScalingPolicies。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **DescribeScheduledActions** 與 CLI

下列程式碼範例示範如何使用 DescribeScheduledActions。

CLI

AWS CLI

描述排程動作

下列describe-scheduled-actions範例顯示指定服務命名空間之排程動作的詳細資訊：

```
aws application-autoscaling describe-scheduled-actions \
    --service-namespace dynamodb
```

輸出：

```
{
    "ScheduledActions": [
        {
            "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
            "Schedule": "at(2019-05-20T18:35:00)",
            "ResourceId": "table/my-table",
            "CreationTime": 1561571888.361,
            "ScheduledActionARN": "arn:aws:autoscaling:us-
west-2:123456789012:scheduledAction:2d36aa3b-cdf9-4565-
b290-81db519b227d:resource/dynamodb/table/my-table:scheduledActionName/my-first-
scheduled-action",
            "ScalableTargetAction": {
                "MinCapacity": 15,
                "MaxCapacity": 20
            },
            "ScheduledActionName": "my-first-scheduled-action",
            "ServiceNamespace": "dynamodb"
        },
        {
            "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
            "Schedule": "at(2019-05-20T18:40:00)",
            "ResourceId": "table/my-table",
            "CreationTime": 1561571946.021,
            "ScheduledActionARN": "arn:aws:autoscaling:us-
west-2:123456789012:scheduledAction:2d36aa3b-cdf9-4565-
```

```
b290-81db519b227d:resource/dynamodb/table/my-table:scheduledActionName/my-second-
scheduled-action",
            "ScalableTargetAction": {
                "MinCapacity": 5,
                "MaxCapacity": 10
            },
            "ScheduledActionName": "my-second-scheduled-action",
            "ServiceNamespace": "dynamodb"
        }
    ]
}
```

如需詳細資訊，請參閱《應用程式自動擴展使用者指南》中的排程擴展。

- 如需 API 詳細資訊，請參閱《 AWS CLI 命令參考》中的 DescribeScheduledActions。

PowerShell

PowerShell V4 的工具

範例 1：此 cmdlet 會列出為 Auto Scaling 群組排程且尚未執行或尚未到達結束時間的動作。

```
Get-AASScheduledAction -ServiceNamespace AppStream
```

輸出：

```
CreationTime         : 12/22/2019 9:25:52 AM
EndTime              : 1/1/0001 12:00:00 AM
ResourceId           : fleet/MyFleet
ScalableDimension    : appstream:fleet:DesiredCapacity
ScalableTargetAction : Amazon.ApplicationAutoScaling.Model.ScalableTargetAction
Schedule             : cron(0 0 8 ? * MON-FRI *)
ScheduledActionARN   : arn:aws:autoscaling:us-
west-2:012345678912:scheduledAction:4897ca24-3caa-4bf1-8484-851a089b243c:resource/
appstream/fleet/MyFleet:scheduledActionName
                       /WeekDaysFleetScaling
ScheduledActionName  : WeekDaysFleetScaling
ServiceNamespace     : appstream
StartTime            : 1/1/0001 12:00:00 AM
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet 參考 (V4) 中的 DescribeScheduledActions。

PowerShell V5 的工具

範例 1：此 cmdlet 會列出為 Auto Scaling 群組排程且尚未執行或尚未到達結束時間的動作。

```
Get-AASScheduledAction -ServiceNamespace AppStream
```

輸出：

```
CreationTime         : 12/22/2019 9:25:52 AM
EndTime              : 1/1/0001 12:00:00 AM
ResourceId           : fleet/MyFleet
ScalableDimension    : appstream:fleet:DesiredCapacity
ScalableTargetAction : Amazon.ApplicationAutoScaling.Model.ScalableTargetAction
Schedule             : cron(0 0 8 ? * MON-FRI *)
ScheduledActionARN   : arn:aws:autoscaling:us-
west-2:012345678912:scheduledAction:4897ca24-3caa-4bf1-8484-851a089b243c:resource/
appstream/fleet/MyFleet:scheduledActionName
                       /WeekDaysFleetScaling
ScheduledActionName  : WeekDaysFleetScaling
ServiceNamespace     : appstream
StartTime            : 1/1/0001 12:00:00 AM
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet 參考 (V5) 中的 DescribeScheduledActions。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **PutScalingPolicy** 與 CLI

下列程式碼範例示範如何使用 `PutScalingPolicy`。

CLI

AWS CLI

範例 1：套用具備預先定義指標規格的目標追蹤擴展政策

下列`put-scaling-policy`範例會將具有預先定義指標規格的目標追蹤擴展政策套用至預設叢集中稱為 web-app 的 Amazon ECS 服務。此政策會將服務的平均 CPU 使用率保持在 75%，向

外擴展和向內擴展冷卻時間為 60 秒。輸出包含代表您建立的兩個 CloudWatch 警示的 ARNs 和名稱。

```
aws application-autoscaling put-scaling-policy --service-namespace ecs \
--scalable-dimension ecs:service:DesiredCount \
--resource-id service/default/web-app \
--policy-name cpu75-target-tracking-scaling-policy --policy-
type TargetTrackingScaling \
--target-tracking-scaling-policy-configuration file://config.json
```

此範例假設您在目前目錄中有一個 config.json 檔案，其中包含以下內容：

```
{
     "TargetValue": 75.0,
     "PredefinedMetricSpecification": {
         "PredefinedMetricType": "ECSServiceAverageCPUUtilization"
     },
     "ScaleOutCooldown": 60,
    "ScaleInCooldown": 60
}
```

輸出：

```
{
    "PolicyARN": "arn:aws:autoscaling:us-
west-2:012345678910:scalingPolicy:6d8972f3-efc8-437c-92d1-6270f29a66e7:resource/
ecs/service/default/web-app:policyName/cpu75-target-tracking-scaling-policy",
    "Alarms": [
        {
            "AlarmARN": "arn:aws:cloudwatch:us-
west-2:012345678910:alarm:TargetTracking-service/default/web-app-AlarmHigh-
d4f0770c-b46e-434a-a60f-3b36d653feca",
            "AlarmName": "TargetTracking-service/default/web-app-AlarmHigh-
d4f0770c-b46e-434a-a60f-3b36d653feca"
        },
        {
            "AlarmARN": "arn:aws:cloudwatch:us-
west-2:012345678910:alarm:TargetTracking-service/default/web-app-
AlarmLow-1b437334-d19b-4a63-a812-6c67aaf2910d",
            "AlarmName": "TargetTracking-service/default/web-app-
AlarmLow-1b437334-d19b-4a63-a812-6c67aaf2910d"
        }
```

```
    ]
}
```

範例 2：套用具備自訂指標規格的目標追蹤擴展政策

下列put-scaling-policy範例會將具有自訂指標規格的目標追蹤擴展政策套用至預設叢集中稱為 web-app 的 Amazon ECS 服務。此政策會將服務的平均使用率保持在 75%，向外擴展和向內擴展冷卻時間為 60 秒。輸出包含代表您建立的兩個 CloudWatch 警示的 ARNs 和名稱。

```
aws application-autoscaling put-scaling-policy --service-namespace ecs \
--scalable-dimension ecs:service:DesiredCount \
--resource-id service/default/web-app \
--policy-name cms75-target-tracking-scaling-policy
--policy-type TargetTrackingScaling \
--target-tracking-scaling-policy-configuration file://config.json
```

此範例假設您在目前目錄中有一個 config.json 檔案，其中包含以下內容：

```
{
    "TargetValue":75.0,
    "CustomizedMetricSpecification":{
        "MetricName":"MyUtilizationMetric",
        "Namespace":"MyNamespace",
        "Dimensions": [
            {
                "Name":"MyOptionalMetricDimensionName",
                "Value":"MyOptionalMetricDimensionValue"
            }
        ],
        "Statistic":"Average",
        "Unit":"Percent"
    },
    "ScaleOutCooldown": 60,
    "ScaleInCooldown": 60
}
```

輸出：

```
{
    "PolicyARN": "arn:aws:autoscaling:us-west-2:012345678910:scalingPolicy:
 8784a896-b2ba-47a1-b08c-27301cc499a1:resource/ecs/service/default/web-
app:policyName/cms75-target-tracking-scaling-policy",
```

```
    "Alarms": [
        {
            "AlarmARN": "arn:aws:cloudwatch:us-
west-2:012345678910:alarm:TargetTracking-service/default/web-app-
AlarmHigh-9bc77b56-0571-4276-ba0f-d4178882e0a0",
            "AlarmName": "TargetTracking-service/default/web-app-
AlarmHigh-9bc77b56-0571-4276-ba0f-d4178882e0a0"
        },
        {
            "AlarmARN": "arn:aws:cloudwatch:us-
west-2:012345678910:alarm:TargetTracking-service/default/web-app-
AlarmLow-9b6ad934-6d37-438e-9e05-02836ddcbdc4",
            "AlarmName": "TargetTracking-service/default/web-app-
AlarmLow-9b6ad934-6d37-438e-9e05-02836ddcbdc4"
        }
    ]
}
```

範例 3：只針對擴增套用目標追蹤擴展政策

下列put-scaling-policy範例會將目標追蹤擴展政策套用至預設叢集web-app中呼叫的 Amazon ECS 服務。當 Application Load Balancer RequestCountPerTarget 的指標超過閾值時，此政策會用來向外擴展 ECS 服務。輸出包含代表您建立的 CloudWatch 警示的 ARN 和名稱。

```
aws application-autoscaling put-scaling-policy \
    --service-namespace ecs \
    --scalable-dimension ecs:service:DesiredCount \
    --resource-id service/default/web-app \
    --policy-name alb-scale-out-target-tracking-scaling-policy \
    --policy-type TargetTrackingScaling \
    --target-tracking-scaling-policy-configuration file://config.json
```

config.json 的內容：

```
{
     "TargetValue": 1000.0,
     "PredefinedMetricSpecification": {
         "PredefinedMetricType": "ALBRequestCountPerTarget",
         "ResourceLabel": "app/EC2Co-EcsEl-1TKLTMITMM0EO/f37c06a68c1748aa/
targetgroup/EC2Co-Defau-LDNM7Q3ZH1ZN/6d4ea56ca2d6a18d"
     },
```

```
     "ScaleOutCooldown": 60,
    "ScaleInCooldown": 60,
    "DisableScaleIn": true
}
```

輸出：

```
{
    "PolicyARN": "arn:aws:autoscaling:us-
west-2:123456789012:scalingPolicy:6d8972f3-efc8-437c-92d1-6270f29a66e7:resource/
ecs/service/default/web-app:policyName/alb-scale-out-target-tracking-scaling-
policy",
    "Alarms": [
        {
            "AlarmName": "TargetTracking-service/default/web-app-AlarmHigh-
d4f0770c-b46e-434a-a60f-3b36d653feca",
            "AlarmARN": "arn:aws:cloudwatch:us-
west-2:123456789012:alarm:TargetTracking-service/default/web-app-AlarmHigh-
d4f0770c-b46e-434a-a60f-3b36d653feca"
        }
    ]
}
```

如需詳細資訊，請參閱《Application Auto Scaling 使用者指南》中的 Application Auto Scaling 的目標追蹤擴展政策。 AWS Auto Scaling

- 如需 API 詳細資訊，請參閱《 AWS CLI 命令參考》中的 PutScalingPolicy。

PowerShell

PowerShell V4 的工具

範例 1：此 cmdlet 會建立或更新 Application Auto Scaling 可擴展目標的政策。每個可擴展的目標都由服務命名空間、資源 ID 和可擴展維度識別。

```
Set-AASScalingPolicy -ServiceNamespace AppStream -PolicyName ASFleetScaleInPolicy
 -PolicyType StepScaling  -ResourceId fleet/MyFleet -ScalableDimension
 appstream:fleet:DesiredCapacity -StepScalingPolicyConfiguration_AdjustmentType
 ChangeInCapacity -StepScalingPolicyConfiguration_Cooldown 360
 -StepScalingPolicyConfiguration_MetricAggregationType Average -
StepScalingPolicyConfiguration_StepAdjustments @{ScalingAdjustment = -1;
 MetricIntervalUpperBound = 0}
```

輸出：

```
Alarms    PolicyARN
------    ---------
{}        arn:aws:autoscaling:us-
west-2:012345678912:scalingPolicy:4897ca24-3caa-4bf1-8484-851a089b243c:resource/
appstream/fleet/MyFleet:policyName/ASFleetScaleInPolicy
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet 參考 (V4) 中的 PutScalingPolicy。

PowerShell V5 的工具

範例 1：此 cmdlet 會建立或更新 Application Auto Scaling 可擴展目標的政策。每個可擴展的目標都由服務命名空間、資源 ID 和可擴展維度識別。

```
Set-AASScalingPolicy -ServiceNamespace AppStream -PolicyName ASFleetScaleInPolicy
 -PolicyType StepScaling  -ResourceId fleet/MyFleet -ScalableDimension
 appstream:fleet:DesiredCapacity -StepScalingPolicyConfiguration_AdjustmentType
 ChangeInCapacity -StepScalingPolicyConfiguration_Cooldown 360
 -StepScalingPolicyConfiguration_MetricAggregationType Average -
StepScalingPolicyConfiguration_StepAdjustments @{ScalingAdjustment = -1;
 MetricIntervalUpperBound = 0}
```

輸出：

```
Alarms    PolicyARN
------    ---------
{}        arn:aws:autoscaling:us-
west-2:012345678912:scalingPolicy:4897ca24-3caa-4bf1-8484-851a089b243c:resource/
appstream/fleet/MyFleet:policyName/ASFleetScaleInPolicy
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet 參考 (V5) 中的 PutScalingPolicy。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

搭配使用 **PutScheduledAction** 與 CLI

下列程式碼範例示範如何使用 PutScheduledAction。

CLI

AWS CLI

將排程動作新增至 DynamoDB 資料表

此範例會將排程動作新增至名為 TestTable 的 DynamoDB 資料表，以根據週期性排程進行擴展。根據指定的排程（每天 12：15pm UTC)，如果目前容量低於為 MinCapacity 指定的值，Application Auto Scaling 會向外擴展至 MinCapacity 指定的值。

命令：

```
aws application-autoscaling put-scheduled-action --service-
namespace dynamodb --scheduled-action-name my-recurring-action --
schedule "cron(15 12 * * ? *)" --resource-id table/TestTable --
scalable-dimension dynamodb:table:WriteCapacityUnits --scalable-target-
action MinCapacity=6
```

如需詳細資訊，請參閱《Application Auto Scaling 使用者指南》中的排程擴展。

- 如需 API 詳細資訊，請參閱《 AWS CLI 命令參考》中的 PutScheduledAction。

PowerShell

PowerShell V4 的工具

範例 1：此 cmdlet 會建立或更新 Application Auto Scaling 可擴展目標的排程動作。每個可擴展的目標都由服務命名空間、資源 ID 和可擴展維度識別。

```
Set-AASScheduledAction -ServiceNamespace AppStream -ResourceId fleet/
MyFleet -Schedule "cron(0 0 8 ? * MON-FRI *)" -ScalableDimension
 appstream:fleet:DesiredCapacity -ScheduledActionName WeekDaysFleetScaling -
ScalableTargetAction_MinCapacity 5 -ScalableTargetAction_MaxCapacity 10
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V4)》中的 PutScheduledAction。

PowerShell V5 的工具

範例 1：此 cmdlet 會建立或更新 Application Auto Scaling 可擴展目標的排程動作。每個可擴展的目標都由服務命名空間、資源 ID 和可擴展維度識別。

```
Set-AASScheduledAction -ServiceNamespace AppStream -ResourceId fleet/
MyFleet -Schedule "cron(0 0 8 ? * MON-FRI *)" -ScalableDimension
 appstream:fleet:DesiredCapacity -ScheduledActionName WeekDaysFleetScaling -
ScalableTargetAction_MinCapacity 5 -ScalableTargetAction_MaxCapacity 10
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V5)》中的 PutScheduledAction。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

`RegisterScalableTarget` 搭配 AWS SDK 或 CLI 使用

下列程式碼範例示範如何使用 `RegisterScalableTarget`。

CLI

AWS CLI

範例 1：將 ECS 服務註冊為可擴展的目標

下列`register-scalable-target`範例向 Application Auto Scaling 註冊 Amazon ECS 服務。它也會將具有金鑰名稱`environment`和值的標籤新增至`production`可擴展的目標。

```
aws application-autoscaling register-scalable-target \
    --service-namespace ecs \
    --scalable-dimension ecs:service:DesiredCount \
    --resource-id service/default/web-app \
    --min-capacity 1 --max-capacity 10 \
    --tags environment=production
```

輸出：

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:us-
west-2:123456789012:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
}
```

如需其他 AWS 服務和自訂資源的範例，請參閱《Application AWS Auto Scaling 使用者指南》中的可與 Application Auto Scaling 搭配使用之服務中的主題。 Auto Scaling

範例 2：暫停可擴展目標的擴展活動

下列register-scalable-target範例會暫停現有可擴展目標的擴展活動。

```
aws application-autoscaling register-scalable-target \
    --service-namespace dynamodb \
    --scalable-dimension dynamodb:table:ReadCapacityUnits \
    --resource-id table/my-table \
    --suspended-
state DynamicScalingInSuspended=true,DynamicScalingOutSuspended=true,ScheduledScalingSusp
```

輸出：

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:us-
west-2:123456789012:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
}
```

如需詳細資訊，請參閱《Application Auto Scaling 使用者指南》中的暫停和繼續 Application Auto Scaling 的擴展。

範例 3：恢復可擴展目標的擴展活動

下列register-scalable-target範例會繼續現有可擴展目標的擴展活動。

```
aws application-autoscaling register-scalable-target \
    --service-namespace dynamodb \
    --scalable-dimension dynamodb:table:ReadCapacityUnits \
    --resource-id table/my-table \
    --suspended-
state DynamicScalingInSuspended=false,DynamicScalingOutSuspended=false,ScheduledScalingSu
```

輸出：

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:us-
west-2:123456789012:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
}
```

如需詳細資訊，請參閱《Application Auto Scaling 使用者指南》中的暫停和繼續 Application Auto Scaling 的擴展。

- 如需 API 詳細資訊，請參閱《 AWS CLI 命令參考》中的 RegisterScalableTarget。

Java

SDK for Java 2.x

> **Note**
>
> GitHub 上提供更多範例。尋找完整範例，並了解如何在 AWS 程式碼範例儲存庫中設定和執行。

```
import software.amazon.awssdk.regions.Region;
import
 software.amazon.awssdk.services.applicationautoscaling.ApplicationAutoScalingClient;
import
 software.amazon.awssdk.services.applicationautoscaling.model.ApplicationAutoScalingExcep
import
 software.amazon.awssdk.services.applicationautoscaling.model.DescribeScalableTargetsRequ
import
 software.amazon.awssdk.services.applicationautoscaling.model.DescribeScalableTargetsResp
import
 software.amazon.awssdk.services.applicationautoscaling.model.DescribeScalingPoliciesRequ
import
 software.amazon.awssdk.services.applicationautoscaling.model.DescribeScalingPoliciesResp
import software.amazon.awssdk.services.applicationautoscaling.model.PolicyType;
import
 software.amazon.awssdk.services.applicationautoscaling.model.PredefinedMetricSpecificati
import
 software.amazon.awssdk.services.applicationautoscaling.model.PutScalingPolicyRequest;
import
 software.amazon.awssdk.services.applicationautoscaling.model.RegisterScalableTargetReque
import
 software.amazon.awssdk.services.applicationautoscaling.model.ScalingPolicy;
import
 software.amazon.awssdk.services.applicationautoscaling.model.ServiceNamespace;
import
 software.amazon.awssdk.services.applicationautoscaling.model.ScalableDimension;
import software.amazon.awssdk.services.applicationautoscaling.model.MetricType;
import
 software.amazon.awssdk.services.applicationautoscaling.model.TargetTrackingScalingPolicy
import java.util.List;
```

```
/**
 * Before running this Java V2 code example, set up your development environment,
 including your credentials.
 *
 * For more information, see the following documentation topic:
 *
 * https://docs.aws.amazon.com/sdk-for-java/latest/developer-guide/get-
started.html
 */
public class EnableDynamoDBAutoscaling {
    public static void main(String[] args) {
        final String usage = """

            Usage:
               <tableId> <roleARN> <policyName>\s

            Where:
               tableId - The table Id value (for example, table/Music).
               roleARN - The ARN of the role that has ApplicationAutoScaling
 permissions.
               policyName - The name of the policy to create.

            """;

        if (args.length != 3) {
            System.out.println(usage);
            System.exit(1);
        }

        System.out.println("This example registers an Amazon DynamoDB table,
 which is the resource to scale.");
        String tableId = args[0];
        String roleARN = args[1];
        String policyName = args[2];
        ServiceNamespace ns = ServiceNamespace.DYNAMODB;
        ScalableDimension tableWCUs =
 ScalableDimension.DYNAMODB_TABLE_WRITE_CAPACITY_UNITS;
        ApplicationAutoScalingClient appAutoScalingClient =
 ApplicationAutoScalingClient.builder()
            .region(Region.US_EAST_1)
            .build();
```

```
        registerScalableTarget(appAutoScalingClient, tableId, roleARN, ns,
tableWCUs);
        verifyTarget(appAutoScalingClient, tableId, ns, tableWCUs);
        configureScalingPolicy(appAutoScalingClient, tableId, ns, tableWCUs,
policyName);
    }

    public static void registerScalableTarget(ApplicationAutoScalingClient
appAutoScalingClient, String tableId, String roleARN, ServiceNamespace ns,
ScalableDimension tableWCUs) {
        try {
            RegisterScalableTargetRequest targetRequest =
RegisterScalableTargetRequest.builder()
                .serviceNamespace(ns)
                .scalableDimension(tableWCUs)
                .resourceId(tableId)
                .roleARN(roleARN)
                .minCapacity(5)
                .maxCapacity(10)
                .build();

            appAutoScalingClient.registerScalableTarget(targetRequest);
            System.out.println("You have registered " + tableId);

        } catch (ApplicationAutoScalingException e) {
            System.err.println(e.awsErrorDetails().errorMessage());
        }
    }

    // Verify that the target was created.
    public static void verifyTarget(ApplicationAutoScalingClient
appAutoScalingClient, String tableId, ServiceNamespace ns, ScalableDimension
tableWCUs) {
        DescribeScalableTargetsRequest dscRequest =
DescribeScalableTargetsRequest.builder()
            .scalableDimension(tableWCUs)
            .serviceNamespace(ns)
            .resourceIds(tableId)
            .build();

        DescribeScalableTargetsResponse response =
appAutoScalingClient.describeScalableTargets(dscRequest);
        System.out.println("DescribeScalableTargets result: ");
        System.out.println(response);
```

```
   }

   // Configure a scaling policy.
   public static void configureScalingPolicy(ApplicationAutoScalingClient
appAutoScalingClient, String tableId, ServiceNamespace ns, ScalableDimension
tableWCUs, String policyName) {
       // Check if the policy exists before creating a new one.
       DescribeScalingPoliciesResponse describeScalingPoliciesResponse =
appAutoScalingClient.describeScalingPolicies(DescribeScalingPoliciesRequest.builder()
           .serviceNamespace(ns)
           .resourceId(tableId)
           .scalableDimension(tableWCUs)
           .build());

       if (!describeScalingPoliciesResponse.scalingPolicies().isEmpty()) {
           // If policies exist, consider updating an existing policy instead of
creating a new one.
           System.out.println("Policy already exists. Consider updating it
instead.");
           List<ScalingPolicy> polList =
describeScalingPoliciesResponse.scalingPolicies();
           for (ScalingPolicy pol : polList) {
               System.out.println("Policy name:" +pol.policyName());
           }
       } else {
           // If no policies exist, proceed with creating a new policy.
           PredefinedMetricSpecification specification =
PredefinedMetricSpecification.builder()

.predefinedMetricType(MetricType.DYNAMO_DB_WRITE_CAPACITY_UTILIZATION)
               .build();

           TargetTrackingScalingPolicyConfiguration policyConfiguration =
TargetTrackingScalingPolicyConfiguration.builder()
               .predefinedMetricSpecification(specification)
               .targetValue(50.0)
               .scaleInCooldown(60)
               .scaleOutCooldown(60)
               .build();

           PutScalingPolicyRequest putScalingPolicyRequest =
PutScalingPolicyRequest.builder()
               .targetTrackingScalingPolicyConfiguration(policyConfiguration)
               .serviceNamespace(ns)
```

```
                .scalableDimension(tableWCUs)
                .resourceId(tableId)
                .policyName(policyName)
                .policyType(PolicyType.TARGET_TRACKING_SCALING)
                .build();

            try {
                appAutoScalingClient.putScalingPolicy(putScalingPolicyRequest);
                System.out.println("You have successfully created a scaling
 policy for an Application Auto Scaling scalable target");
            } catch (ApplicationAutoScalingException e) {
                System.err.println("Error: " +
 e.awsErrorDetails().errorMessage());
            }
        }
    }
}
```

- 如需 API 詳細資訊，請參閱《 AWS SDK for Java 2.x API 參考》中的 RegisterScalableTarget。

PowerShell

PowerShell V4 的工具

範例 1：此 cmdlet 會註冊或更新可擴展的目標。可擴展的目標是 Application Auto Scaling 可以向外擴展和向內擴展的資源。

```
Add-AASScalableTarget -ServiceNamespace AppStream -ResourceId fleet/MyFleet -
ScalableDimension appstream:fleet:DesiredCapacity -MinCapacity 2 -MaxCapacity 10
```

- 如需 API 詳細資訊，請參閱AWS Tools for PowerShell 《 Cmdlet Reference (V4)》中的 RegisterScalableTarget。

PowerShell V5 的工具

範例 1：此 cmdlet 會註冊或更新可擴展的目標。可擴展的目標是 Application Auto Scaling 可以向外擴展和向內擴展的資源。

```
Add-AASScalableTarget -ServiceNamespace AppStream -ResourceId fleet/MyFleet -
ScalableDimension appstream:fleet:DesiredCapacity -MinCapacity 2 -MaxCapacity 10
```

- 如需 API 詳細資訊，請參閱 AWS Tools for PowerShell Cmdlet Reference (V5) 中的 RegisterScalableTarget。

如需 AWS SDK 開發人員指南和程式碼範例的完整清單，請參閱 搭配 AWS SDK 使用此服務。此主題也包含有關入門的資訊和舊版 SDK 的詳細資訊。

Application Auto Scaling 的標籤支援

您可以使用 AWS CLI 或 SDK 來標記 Application Auto Scaling 可擴展目標。可擴展目標是指代表 Application Auto Scaling 可以擴展之 AWS 或 自訂資源的實體。

每個標籤都是使用 Application Auto Scaling API 的使用者定義索引鍵和值組成的標籤。標籤可以幫助您根據組織需求來精細存取特定的可擴展目標。如需詳細資訊，請參閱帶有 Application Auto Scaling 的 ABAC。

您可以在註冊新的可擴展目標時向其新增標籤，或者可以將標籤新增至現有的可擴展目標。

管理標籤的常用命令包括：

- register-scalable-target，用於在註冊新的可擴展目標時對其進行標記。
- tag-resource，用於將標籤新增至現有的可擴展目標。
- list-tags-for-resource，用於傳回可擴展目標上的標籤。
- untag-resource，用於刪除標籤。

標記範例

使用 register-scalable-target 命令與 `--tags` 選項，如下所示。此範例標記具有兩個標籤的目標：一個名為 **`environment`** 且標籤值為 **`production`** 的標籤索引鍵，以及一個名為 **`iscontainerbased`** 且標籤值為 **`true`** 的標籤索引鍵。

將 `--min-capacity`和 的範例值`--max-capacity`和 的範例文字`--service-namespace`取代為您搭配 Application Auto Scaling 使用 AWS 的服務命名空間、`--scalable-dimension`與您正在註冊的資源相關聯的可擴展維度，以及`--resource-id`與資源的識別符。如需每個服務的詳細資訊和範例，請參閱 AWS 服務 可與 Application Auto Scaling 搭配使用 中的主題。

```
aws application-autoscaling register-scalable-target \
  --service-namespace namespace \
  --scalable-dimension dimension \
  --resource-id identifier \
  --min-capacity 1 --max-capacity 10 \
  --tags environment=production,iscontainerbased=true
```

如果成功，此命令會傳回可擴展目標的 ARN。

```
{
    "ScalableTargetARN": "arn:aws:application-autoscaling:region:account-id:scalable-target/1234abcd56ab78cd901ef1234567890ab123"
}
```

> **Note**
>
> 如果此命令擲回錯誤，請確定您已在 AWS CLI 本機將 更新為最新版本。

安全標籤

使用標籤來確認請求者 (例如 IAM 使用者或角色) 具有執行特定動作的許可。使用下列一個或多個條件金鑰，在 IAM 政策的條件元素中提供標籤資訊：

- 使用 `aws:ResourceTag/tag-key: tag-value` 允許 (或拒絕) 在有特定標籤的可擴展目標上的使用者動作。
- 使用 `aws:RequestTag/tag-key: tag-value` 要求在請求中出現 (或不出現) 特定標籤。
- 使用 `aws:TagKeys [tag-key, ...]` 要求在請求中出現 (或不出現) 特定標籤鍵。

例如，以下 IAM 政策授予許可以使用 DeregisterScalableTarget、DeleteScalingPolicy 和 DeleteScheduledAction 動作。但是，如果被採取行動的可擴展目標有標籤 **environment=production**，它也會拒絕該動作。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "application-autoscaling:DeregisterScalableTarget",
                "application-autoscaling:DeleteScalingPolicy",
                "application-autoscaling:DeleteScheduledAction"
            ],
            "Resource": "*"
            }
        },
        {
            "Effect": "Deny",
```

```
            "Action": [
                "application-autoscaling:DeregisterScalableTarget",
                "application-autoscaling:DeleteScalingPolicy",
                "application-autoscaling:DeleteScheduledAction"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {"aws:ResourceTag/environment": "production"}
            }
        }
    ]
}
```

控制對標籤的存取

使用標籤來確認請求者 (例如 IAM 使用者或角色) 具有新增、修改或刪除可擴展目標之標籤的許可。

例如，您可以建立 IAM 政策，僅允許移除具有來自可擴展目標之 **temporary** 索引鍵的標籤。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "application-autoscaling:UntagResource",
            "Resource": "*",
            "Condition": {
                "ForAllValues:StringEquals": { "aws:TagKeys": ["temporary"] }
            }
        }
    ]
}
```

Application Auto Scaling 中的安全

的雲端安全性 AWS 是最高優先順序。身為 AWS 客戶，您可以受益於資料中心和網路架構，該架構專為滿足最安全敏感組織的需求而建置。

安全性是 AWS 與您之間的共同責任。共同責任模型將其描述為雲端的安全性和雲端中的安全性：

- 雲端的安全性 – AWS 負責保護在 AWS Cloud 中執行 AWS 服務的基礎設施。 AWS 也為您提供可安全使用的服務。在AWS 合規計劃中，第三方稽核人員會定期測試和驗證我們的安全有效性。若要了解適用於 Application Auto Scaling 的合規計劃，請參閱AWS 合規計劃範圍內的服務。
- 雲端的安全性 – 您的責任取決於您使用 AWS 的服務。您也必須對其他因素負責，包括資料的機密性、您公司的要求和適用法律和法規。

本文件有助於您了解如何在使用 Application Auto Scaling 時套用共同的責任模型。下列主題說明如何設定 Application Auto Scaling 來達成安全與合規目標。您也會了解如何使用其他 AWS 服務來協助您監控和保護 Application Auto Scaling 資源。

目錄

Application Auto Scaling 中的資料保護

AWS 共同責任模型適用於 Application Auto Scaling 中的資料保護。如此模型所述， AWS 負責保護執行所有 的全域基礎設施 AWS 雲端。您負責維護在此基礎設施上託管內容的控制權。您也同時負責所使用 AWS 服務 的安全組態和管理任務。如需資料隱私權的詳細資訊，請參閱資料隱私權常見問答集。如需有關歐洲資料保護的相關資訊，請參閱 AWS 安全性部落格上的 AWS 共同的責任模型和 GDPR 部落格文章。

基於資料保護目的，我們建議您保護 AWS 帳戶 登入資料，並使用 AWS IAM Identity Center 或 AWS Identity and Access Management (IAM) 設定個別使用者。如此一來，每個使用者都只會獲得授與完成其任務所必須的許可。我們也建議您採用下列方式保護資料：

- 每個帳戶均要使用多重要素驗證 (MFA)。
- 使用 SSL/TLS 與 AWS 資源通訊。我們需要 TLS 1.2 並建議使用 TLS 1.3。
- 使用 設定 API 和使用者活動記錄 AWS CloudTrail。如需有關使用 CloudTrail 追蹤擷取 AWS 活動的資訊，請參閱AWS CloudTrail 《 使用者指南》中的使用 CloudTrail 追蹤。
- 使用 AWS 加密解決方案，以及其中的所有預設安全控制 AWS 服務。
- 使用進階的受管安全服務 (例如 Amazon Macie)，協助探索和保護儲存在 Amazon S3 的敏感資料。
- 如果您在 AWS 透過命令列界面或 API 存取 時需要 FIPS 140-3 驗證的密碼編譯模組，請使用 FIPS 端點。如需有關 FIPS 和 FIPS 端點的更多相關資訊，請參閱聯邦資訊處理標準 (FIPS) 140-3。

我們強烈建議您絕對不要將客戶的電子郵件地址等機密或敏感資訊，放在標籤或自由格式的文字欄位中，例如名稱欄位。這包括當您使用 Application Auto Scaling 或使用主控台 AWS CLI、API 或 AWS SDKs的其他 AWS 服務 時。您在標籤或自由格式文字欄位中輸入的任何資料都可能用於計費或診斷日誌。如果您提供外部伺服器的 URL，我們強烈建議請勿在驗證您對該伺服器請求的 URL 中包含憑證資訊。

適用於 Application Auto Scaling 的 Identity and Access Management

AWS Identity and Access Management (IAM) 是 AWS 服務 ，可協助管理員安全地控制對 AWS 資源的存取。IAM 管理員可以控制誰能完成身分驗證 (已登入) 和獲得授權 (具有許可) 而得以使用 Application Auto Scaling 資源。IAM 是 AWS 服務 您可以免費使用的 。

如需完整的 IAM 文件，請參閱 IAM 使用者指南。

存取控制

您可以持有效憑證來驗證請求，但還須具備許可，才能建立或存取 Application Auto Scaling 資源。例如，您必須具有建立擴展政策、設定排程擴展等許可。

下列各節提供 IAM 管理員如何使用 IAM 透過控制可執行 Application Auto Scaling API 動作的人員來協助保護 AWS 資源的詳細資訊。

目錄

Application Auto Scaling 如何搭配 IAM 一起使用

> **Note**
>
> 在 2017 年 12 月，Application Auto Scaling 有一項更新，對 Application Auto Scaling 整合式服務啟用數個服務連結角色。使用者需要特定的 IAM 許可「和」Application Auto Scaling 服務連結角色 (或 Amazon EMR 自動擴展的服務角色)，才能設定擴展。

在使用 IAM 來管理對 Application Auto Scaling 的存取權之前，請先了解哪些 IAM 功能可以與 Application Auto Scaling 搭配使用。

可以與 Application Auto Scaling 搭配使用的 IAM 功能

IAM 功能	應用程式自動調整規模支援
身分型政策	是
政策動作	是
政策資源	是
政策條件索引鍵 (服務特定)	是
資源型政策	否
ACL	否
ABAC(政策中的標籤)	部分

IAM 功能	應用程式自動調整規模支援
臨時憑證	是
服務角色	是
服務連結角色	是

若要全面了解 Application Auto Scaling 和其他 如何與大多數 IAM 功能 AWS 服務 搭配使用，請參閱《AWS 服務 IAM 使用者指南》中的 與 IAM 搭配使用。

Application Auto Scaling 以身分為基礎的政策

支援身分型政策：是

身分型政策是可以附加到身分 (例如 IAM 使用者、使用者群組或角色) 的 JSON 許可政策文件。這些政策可控制身分在何種條件下能對哪些資源執行哪些動作。如需了解如何建立身分型政策，請參閱《IAM 使用者指南》中的透過客戶管理政策定義自訂 IAM 許可。

使用 IAM 身分型政策，您可以指定允許或拒絕的動作和資源，以及在何種條件下允許或拒絕動作。您無法在身分型政策中指定主體，因為這會套用至連接的使用者或角色。如要了解您在 JSON 政策中使用的所有元素，請參閱《IAM 使用者指南》中的 IAM JSON 政策元素參考。

Application Auto Scaling 的身分型政策範例

若要檢視 Application Auto Scaling 以身分為基礎的政策範例，請參閱 Application Auto Scaling 以身分為基礎的政策範例。

動作

支援政策動作：是

在 IAM 政策陳述式中，您可以從任何支援 IAM 的服務指定任何 API 動作。針對 Application Auto Scaling，請在 API 動作名稱使用下列字首：`application-autoscaling:`。例如：`application-autoscaling:RegisterScalableTarget`、`application-autoscaling:PutScalingPolicy` 和 `application-autoscaling:DeregisterScalableTarget`。

若要在單一陳述式中指定多個動作，請以逗號分隔它們，如下列範例所示。

```
"Action": [
```

```
        "application-autoscaling:DescribeScalingPolicies",
        "application-autoscaling:DescribeScalingActivities"
```

您也可以使用萬用字元 (*) 來指定多個動作。例如，如需指定開頭是 Describe 文字的所有動作，請包含以下動作：

```
"Action": "application-autoscaling:Describe*"
```

如需 Application Auto Scaling 動作的清單，請參閱《服務授權參考》中的 AWS Application Auto Scaling 定義的動作。

資源

支援政策資源：是

在 IAM 政策陳述式中，Resource 元素指定陳述式所涵蓋的一個或多個物件。對於 Application Auto Scaling，每個 IAM 政策陳述式都會套用至您使用其 Amazon Resource Names (ARN) 指定的可擴展目標。

可擴展目標的 ARN 資源格式：

```
arn:aws:application-autoscaling:region:account-id:scalable-target/unique-identifier
```

例如，您可以在陳述式中使用其 ARN 指定特定的可擴展目標，如下所示。唯一 ID (1234abcd56ab78cd901ef1234567890ab123) 是由 Application Auto Scaling 指派給可擴展目標的值。

```
"Resource": "arn:aws:application-autoscaling:us-east-1:123456789012:scalable-
target/1234abcd56ab78cd901ef1234567890ab123"
```

您也可以使用萬用字元 (*) 取代唯一標識符，以此指定所有屬於特定帳戶的執行個體，如下所示。

```
"Resource": "arn:aws:application-autoscaling:us-east-1:123456789012:scalable-target/*"
```

若要指定所有資源，或如果特定的 API 動作不支援 ARN，請使用萬用字元 (*) 作為 Resource 元素，如下所示。

```
"Resource": "*"
```

如需詳細資訊，請參閱《服務授權參考》中的 AWS Application Auto Scaling 定義的資源類型。

條件索引鍵

支援服務特定政策條件金鑰：是

您可以在 IAM 政策中指定條件，這些政策可以控制存取 Application Auto Scaling 資源。政策陳述式只有在符合下列條件時才有效。

Application Auto Scaling 支援下列服務定義條件金鑰，您可以在以身分為基礎的政策中使用這些金鑰來確定誰可以執行 Application Auto Scaling API 動作。

- `application-autoscaling:scalable-dimension`
- `application-autoscaling:service-namespace`

若要了解您可以使用條件索引鍵的 Application Auto Scaling API 動作，請參閱《服務授權參考》中的 AWS Application Auto Scaling 定義的動作。如需使用 Application Auto Scaling 條件索引鍵的詳細資訊，請參閱 AWS Application Auto Scaling 的條件索引鍵。

若要檢視所有服務都可使用的全域條件金鑰，請參閱 IAM 使用者指南中的 AWS 全域條件金鑰。

資源型政策

支援資源型政策：否

其他 AWS 服務，例如 Amazon Simple Storage Service，支援以資源為基礎的許可政策。例如，您可以將許可政策連接至 S3 儲存貯體，以管理該儲存貯體的存取許可。

Application Auto Scaling 不支援以資源為基礎的政策。

存取控制清單 (ACL)

支援 ACL：否

Application Auto Scaling 不支援存取控制清單 (ACL)。

帶有 Application Auto Scaling 的 ABAC

支援 ABAC (政策中的標籤)：部分

屬性型存取控制 (ABAC) 是一種授權策略，可根據屬性來定義許可。在 中 AWS，這些屬性稱為標籤。您可以將標籤連接到 IAM 實體 （使用者或角色） 和許多 AWS 資源。為實體和資源加上標籤是 ABAC 的第一步。您接著要設計 ABAC 政策，允許在主體的標籤與其嘗試存取的資源標籤相符時操作。

ABAC 在成長快速的環境中相當有幫助，並能在政策管理變得繁瑣時提供協助。

如需根據標籤控制存取，請使用 `aws:ResourceTag/key-name`、`aws:RequestTag/key-name` 或 `aws:TagKeys` 條件索引鍵，在政策的條件元素中，提供標籤資訊。

ABAC 可用於支援標籤的資源，但並非所有支援標籤的資源。排程動作和擴展政策不支援標籤，但可擴展目標支援標籤。如需詳細資訊，請參閱Application Auto Scaling 的標籤支援。

如需 ABAC 的詳細資訊，請參閱 IAM 使用者指南中的什麼是 ABAC?。如要查看含有設定 ABAC 步驟的教學課程，請參閱 IAM 使用者指南中的使用屬性型存取控制 (ABAC)。

在 Application Auto Scaling 中使用臨時憑證

支援臨時憑證：是

當您使用臨時登入資料登入時，有些 AWS 服務 無法運作。如需其他資訊，包括哪些 AWS 服務 使用臨時登入資料，請參閱《AWS 服務 IAM 使用者指南》中的 使用 IAM。

如果您 AWS Management Console 使用使用者名稱和密碼以外的任何方法登入 ，則會使用臨時登入資料。例如，當您 AWS 使用公司的單一登入 (SSO) 連結存取 時，該程序會自動建立臨時登入資料。當您以使用者身分登入主控台，然後切換角色時，也會自動建立臨時憑證。如需切換角色的詳細資訊，請參閱《IAM 使用者指南》中的從使用者切換至 IAM 角色 (主控台)。

您可以使用 AWS CLI 或 AWS API 手動建立臨時登入資料。然後，您可以使用這些臨時登入資料來存取 AWS。 AWS 建議您動態產生臨時登入資料，而不是使用長期存取金鑰。如需詳細資訊，請參閱 IAM 中的暫時性安全憑證。

服務角色

支援服務角色：是

如果您的 Amazon EMR 叢集使用自動擴展，則此功能可讓 Application Auto Scaling 代表您擔任服務角色。與服務連結角色類似，服務角色可讓服務存取其他服務中的資源，以代表您完成動作。服務角色會出現在您的 IAM 帳戶中，且由該帳戶所擁有。這表示 IAM 管理員可以變更此角色的許可。不過，這樣可能會破壞此服務的功能。

Application Auto Scaling 只對 Amazon EMR 支援服務角色。如需 EMR 服務角色的文件，請參閱《Amazon EMR 管理指南》中的對執行個體群組搭配自訂政策來使用自動擴展。

> **Note**
>
> 隨著服務連結角色的推出，許多舊式服務角色的功能將被取代，例如 Amazon ECS 和 Spot 機群。

服務連結角色

支援服務連結角色：是

服務連結角色是連結至 的一種服務角色 AWS 服務。服務可以擔任代表您執行動作的角色。服務連結角色會出現在您的 中 AWS 帳戶 ，並由服務擁有。IAM 管理員可以檢視，但不能編輯服務連結角色的許可。

如需 Application Auto Scaling 服務連結角色的相關資訊，請參閱 Application Auto Scaling 的服務連結角色。

AWS Application Auto Scaling 的 受管政策

AWS 受管政策是由 AWS AWS 受管政策建立和管理的獨立政策旨在為許多常用案例提供許可，以便您可以開始將許可指派給使用者、群組和角色。

請記住， AWS 受管政策可能不會授予特定使用案例的最低權限許可，因為這些許可可供所有 AWS 客戶使用。我們建議您定義使用案例專屬的客戶管理政策，以便進一步減少許可。

您無法變更 AWS 受管政策中定義的許可。如果 AWS 更新 AWS 受管政策中定義的許可，則更新會影響政策連接的所有委託人身分（使用者、群組和角色）。當新的 AWS 服務 啟動或新的 API 操作可用於現有服務時， AWS 最有可能更新 AWS 受管政策。

如需詳細資訊，請參閱《IAM 使用者指南》中的 AWS 受管政策。

AWS 受管政策：AppStream 2.0 和 CloudWatch

政策名稱：AWSApplicationAutoscalingAppStreamFleetPolicy

此政策會連接到名為 AWSServiceRoleForApplicationAutoScaling_AppStreamFleet 的服務連結角色，以允許 Application Auto Scaling 呼叫 Amazon AppStream 和 CloudWatch 並代表您執行擴展。

許可詳細資訊

許可政策允許 Application Auto Scaling 針對所有相關資源 ("Resource"： "*") 完成下列動作：

- 動作：`appstream:DescribeFleets`
- 動作：`appstream:UpdateFleet`
- 動作：`cloudwatch:DescribeAlarms`
- 動作：`cloudwatch:PutMetricAlarm`

- 動作：`cloudwatch:DeleteAlarms`

AWS 受管政策：Aurora 和 CloudWatch

政策名稱：AWSApplicationAutoscalingRDSClusterPolicy

此政策會連接到名為 AWSServiceRoleForApplicationAutoScaling_RDSCluster 的服務連結角色，以允許 Application Auto Scaling 呼叫 Aurora 和 CloudWatch 並代表您執行擴展。

許可詳細資訊

許可政策允許 Application Auto Scaling 針對所有相關資源 ("Resource"： "*") 完成下列動作：

- 動作：`rds:AddTagsToResource`
- 動作：`rds:CreateDBInstance`
- 動作：`rds:DeleteDBInstance`
- 動作：`rds:DescribeDBClusters`
- 動作：`rds:DescribeDBInstance`
- 動作：`cloudwatch:DescribeAlarms`
- 動作：`cloudwatch:PutMetricAlarm`
- 動作：`cloudwatch:DeleteAlarms`

AWS 受管政策：Amazon Comprehend 和 CloudWatch

政策名稱：AWSApplicationAutoscalingComprehendEndpointPolicy

此政策會連接到名為 AWSServiceRoleForApplicationAutoScaling_ComprehendEndpoint 的服務連結角色，以允許 Application Auto Scaling 呼叫 Amazon Comprehend 和 CloudWatch 並代表您執行擴展。

許可詳細資訊

許可政策允許 Application Auto Scaling 針對所有相關資源 ("Resource"： "*") 完成下列動作：

- 動作：`comprehend:UpdateEndpoint`
- 動作：`comprehend:DescribeEndpoint`
- 動作：`cloudwatch:DescribeAlarms`

- 動作：cloudwatch:PutMetricAlarm
- 動作：cloudwatch:DeleteAlarms

AWS 受管政策：DynamoDB 和 CloudWatch

政策名稱：AWSApplicationAutoscalingDynamoDBTablePolicy

此政策會連接到名為 AWSServiceRoleForApplicationAutoScaling_DynamoDBTable 的服務連結角色，以允許 Application Auto Scaling 呼叫 DynamoDBand CloudWatch 並代表您執行擴展。

許可詳細資訊

許可政策允許 Application Auto Scaling 針對所有相關資源 ("Resource"： "*") 完成下列動作：

- 動作：dynamodb:DescribeTable
- 動作：dynamodb:UpdateTable
- 動作：cloudwatch:DescribeAlarms
- 動作：cloudwatch:PutMetricAlarm
- 動作：cloudwatch:DeleteAlarms

AWS 受管政策：Amazon ECS 和 CloudWatch

政策名稱：AWSApplicationAutoscalingECSServicePolicy

此政策會連接到名為 AWSServiceRoleForApplicationAutoScaling_ECSService 的服務連結角色，以允許 Application Auto Scaling 呼叫 Amazon ECS 和 CloudWatch 並代表您執行擴展。

許可詳細資訊

許可政策允許 Application Auto Scaling 針對所有相關資源 ("Resource"： "*") 完成下列動作：

- 動作：ecs:DescribeServices
- 動作：ecs:UpdateService
- 動作：cloudwatch:PutMetricAlarm
- 動作：cloudwatch:DescribeAlarms
- 動作：cloudwatch:GetMetricData
- 動作：cloudwatch:DeleteAlarms

AWS 受管政策：ElastiCache 和 CloudWatch

政策名稱：AWSApplicationAutoscalingElastiCacheRGPolicy

此政策會連接到名為 AWSServiceRoleForApplicationAutoScaling_ElastiCacheRG 的服務連結角色，以允許 Application Auto Scaling 呼叫 ElastiCache 和 CloudWatch 並代表您執行擴展。此服務連結角色可用於 ElastiCache Memcached、Redis OSS 和 Valkey。

許可詳細資訊

許可政策允許 Application Auto Scaling 對指定的資源完成下列動作：

- 動作：對所有資源執行 `elasticache:DescribeReplicationGroups`
- 動作：對所有資源執行 `elasticache:ModifyReplicationGroupShardConfiguration`
- 動作：對所有資源執行 `elasticache:IncreaseReplicaCount`
- 動作：對所有資源執行 `elasticache:DecreaseReplicaCount`
- 動作：對所有資源執行 `elasticache:DescribeCacheClusters`
- 動作：對所有資源執行 `elasticache:DescribeCacheParameters`
- 動作：對所有資源執行 `elasticache:ModifyCacheCluster`
- 動作：對資源 `arn:aws:cloudwatch:*:*:alarm:*` 執行 `cloudwatch:DescribeAlarms`
- 動作：對資源 `arn:aws:cloudwatch:*:*:alarm:TargetTracking*` 執行 `cloudwatch:PutMetricAlarm`
- 動作：對資源 `arn:aws:cloudwatch:*:*:alarm:TargetTracking*` 執行 `cloudwatch:DeleteAlarms`

AWS 受管政策：Amazon Keyspaces 和 CloudWatch

政策名稱：AWSApplicationAutoscalingCassandraTablePolicy

此政策會連接到名為 AWSServiceRoleForApplicationAutoScaling_CassandraTable 的服務連結角色，以允許 Application Auto Scaling 呼叫 Amazon Keyspaces 和 CloudWatch 並代表您執行擴展。

許可詳細資訊

許可政策允許 Application Auto Scaling 對指定的資源完成下列動作：

- 動作：在下列資源`cassandra:Select`上：

  - arn:*:cassandra:*:*:/keyspace/system/table/*
  - arn:*:cassandra:*:*:/keyspace/system_schema/table/*
  - arn:*:cassandra:*:*:/keyspace/system_schema_mcs/table/*
- 動作：對所有資源執行 cassandra:Alter
- 動作：對所有資源執行 cloudwatch:DescribeAlarms
- 動作：對所有資源執行 cloudwatch:PutMetricAlarm
- 動作：對所有資源執行 cloudwatch:DeleteAlarms

AWS 受管政策：Lambda 和 CloudWatch

政策名稱：AWSApplicationAutoscalingLambdaConcurrencyPolicy

此政策會連接到名為 AWSServiceRoleForApplicationAutoScaling_LambdaConcurrency 的服務連結角色，以允許 Application Auto Scaling 呼叫 Lambda 和 CloudWatch 並代表您執行擴展。

許可詳細資訊

許可政策允許 Application Auto Scaling 對所有相關資源 ("Resource"： "*") 完成下列動作：

- 動作：lambda:PutProvisionedConcurrencyConfig
- 動作：lambda:GetProvisionedConcurrencyConfig
- 動作：lambda:DeleteProvisionedConcurrencyConfig
- 動作：cloudwatch:DescribeAlarms
- 動作：cloudwatch:PutMetricAlarm
- 動作：cloudwatch:DeleteAlarms

AWS 受管政策：Amazon MSK 和 CloudWatch

政策名稱：AWSApplicationAutoscalingKafkaClusterPolicy

此政策會連接到名為 AWSServiceRoleForApplicationAutoScaling_KafkaCluster 的服務連結角色，以允許 Application Auto Scaling 呼叫 Amazon MSK 和 CloudWatch 並代表您執行擴展。

許可詳細資訊

許可政策允許 Application Auto Scaling 對所有相關資源 ("Resource"： "*") 完成下列動作：

- 動作：kafka:DescribeCluster
- 動作：kafka:DescribeClusterOperation
- 動作：kafka:UpdateBrokerStorage
- 動作：cloudwatch:DescribeAlarms
- 動作：cloudwatch:PutMetricAlarm
- 動作：cloudwatch:DeleteAlarms

AWS 受管政策：Neptune 和 CloudWatch

政策名稱：AWSApplicationAutoscalingNeptuneClusterPolicy

此政策會連接到名為 AWSServiceRoleForApplicationAutoScaling_NeptuneCluster 的服務連結角色，以允許 Application Auto Scaling 呼叫 Neptune 和 CloudWatch 並代表您執行擴展。

許可詳細資訊

許可政策允許 Application Auto Scaling 對指定的資源完成下列動作：

- 動作：對所有資源執行 rds:ListTagsForResource
- 動作：對所有資源執行 rds:DescribeDBInstances
- 動作：對所有資源執行 rds:DescribeDBClusters
- 動作：對所有資源執行 rds:DescribeDBClusterParameters
- 動作：對所有資源執行 cloudwatch:DescribeAlarms
- 動作：對 Amazon Neptune 資料庫引擎 ("Condition":{"StringEquals": {"rds:DatabaseEngine":"neptune"}) 中具有 autoscaled-reader 字首的資源執行 rds:AddTagsToResource
- 動作：對 Amazon Neptune 資料庫引擎 ("Condition":{"StringEquals": {"rds:DatabaseEngine":"neptune"}) 所有資料庫叢集 ("Resource":"arn:*:rds:*:*:db:autoscaled-reader*", "arn:aws:rds:*:*:cluster:*") 中具有 autoscaled-reader 字首的資源執行 rds:CreateDBInstance
- 動作：對資源 arn:aws:rds:*:*:db:autoscaled-reader* 執行 rds:DeleteDBInstance
- 動作：對資源 arn:aws:cloudwatch:*:*:alarm:TargetTracking* 執行 cloudwatch:PutMetricAlarm

- 動作：對資源 `arn:aws:cloudwatch:*:*:alarm:TargetTracking*` 執行 `cloudwatch:DeleteAlarms`

AWS 受管政策：SageMaker AI 和 CloudWatch

政策名稱：AWSApplicationAutoscalingSageMakerEndpointPolicy

此政策會連接到名為 AWSServiceRoleForApplicationAutoScaling_SageMakerEndpoint 的服務連結角色，以允許 Application Auto Scaling 呼叫 SageMaker AI 和 CloudWatch 並代表您執行擴展。

許可詳細資訊

許可政策允許 Application Auto Scaling 對指定的資源完成下列動作：

- 動作：對所有資源執行 `sagemaker:DescribeEndpoint`
- 動作：對所有資源執行 `sagemaker:DescribeEndpointConfig`
- 動作：對所有資源執行 `sagemaker:DescribeInferenceComponent`
- 動作：對所有資源執行 `sagemaker:UpdateEndpointWeightsAndCapacities`
- 動作：對所有資源執行 `sagemaker:UpdateInferenceComponentRuntimeConfig`
- 動作：對所有資源執行 `cloudwatch:DescribeAlarms`
- 動作：對所有資源執行 `cloudwatch:GetMetricData`
- 動作：對資源 `arn:aws:cloudwatch:*:*:alarm:TargetTracking*` 執行 `cloudwatch:PutMetricAlarm`
- 動作：對資源 `arn:aws:cloudwatch:*:*:alarm:TargetTracking*` 執行 `cloudwatch:DeleteAlarms`

AWS 受管政策：EC2 Spot Fleet 和 CloudWatch

政策名稱：AWSApplicationAutoscalingEC2SpotFleetRequestPolicy

此政策會連接到名為 AWSServiceRoleForApplicationAutoScaling_EC2SpotFleetRequest 的服務連結角色，以允許 Application Auto Scaling 呼叫 Amazon EC2 和 CloudWatch 並代表您執行擴展。

許可詳細資訊

許可政策允許 Application Auto Scaling 對所有相關資源 ("Resource"： "*") 完成下列動作：

- 動作：ec2:DescribeSpotFleetRequests
- 動作：ec2:ModifySpotFleetRequest
- 動作：cloudwatch:DescribeAlarms
- 動作：cloudwatch:PutMetricAlarm
- 動作：cloudwatch:DeleteAlarms

AWS 受管政策：WorkSpaces 和 CloudWatch

政策名稱：AWSApplicationAutoscalingWorkSpacesPoolPolicy

此政策會連接到名為 AWSServiceRoleForApplicationAutoScaling_WorkSpacesPool 的服務連結角色，以允許 Application Auto Scaling 呼叫 WorkSpaces 和 CloudWatch 並代表您執行擴展。

許可詳細資訊

許可政策允許 Application Auto Scaling 對指定的資源完成下列動作：

- 動作：在與 SLR 相同帳戶的所有資源workspaces:DescribeWorkspacesPools上
- 動作：在與 SLR 相同帳戶的所有資源workspaces:UpdateWorkspacesPool上
- 動作：cloudwatch:DescribeAlarms在與 SLR 相同帳戶的所有警示上
- 動作：cloudwatch:PutMetricAlarm在與 SLR 相同帳戶的所有警示上，警示名稱開頭為 TargetTracking
- 動作：cloudwatch:DeleteAlarms在與 SLR 相同帳戶的所有警示上，警示名稱開頭為 TargetTracking

AWS 受管政策：自訂資源和 CloudWatch

政策名稱：AWSApplicationAutoScalingCustomResourcePolicy

此政策會連接到名為 AWSServiceRoleForApplicationAutoScaling_CustomResource 的服務連結角色，以允許 Application Auto Scaling 呼叫可透過 API Gateway 和 CloudWatch 使用的自訂資源，並代表您執行擴展。

許可詳細資訊

許可政策允許 Application Auto Scaling 對所有相關資源 ("Resource"： "*") 完成下列動作：

- 動作：`execute-api:Invoke`
- 動作：`cloudwatch:DescribeAlarms`
- 動作：`cloudwatch:PutMetricAlarm`
- 動作：`cloudwatch:DeleteAlarms`

AWS 受管政策的 Application Auto Scaling 更新

檢視自此服務開始追蹤 Application Auto Scaling AWS 受管政策更新以來的詳細資訊。如需有關此頁面變更的自動提醒，請訂閱 Application Auto Scaling Document history (Application Auto Scaling 文件歷程記錄) 頁面上的 RSS 摘要。

變更	描述	日期
AWSApplicationAutoscalingElastiCacheRGPolicy – 更新現有政策	新增呼叫 ElastiCache `ModifyCacheCluster` API 動作以支援 Memcached 自動擴展的許可。	2025 年 4 月 10 日
AWSApplicationAutoscalingECSServicePolicy – 更新現有政策	新增呼叫 CloudWatch `GetMetricData` API 動作以支援預測擴展的許可。	2024 年 11 月 21 日
AWSApplicationAutoscalingWorkSpacesPoolPolicy – 新政策	新增 Amazon WorkSpaces 的受管政策。此政策會連接到服務連結角色，允許 Application Auto Scaling 呼叫 WorkSpaces 和 CloudWatch，並代表您執行擴展。	2024 年 6 月 24 日
AWSApplicationAutoscalingSageMakerEndpointPolicy – 更新至現有政策	新增呼叫 SageMaker AI `DescribeInferenceComponent` 和 `UpdateInferenceComponentRuntimeConfig` API 動作的許可，以支援 SageMaker AI 資源自動擴展的相容性，以供即將進行的整合使用。	2023 年 11 月 13 日

變更	描述	日期
	該政策現在還將 CloudWatch `PutMetricAlarm` 和 `DeleteAlarms` API 動作限制為與目標追蹤擴展政策搭配使用的 CloudWatch 警示。	
AWSApplicationAutoscalingNeptuneClusterPolicy – 新政策	新增 Neptune 的受管政策。此政策會連接到服務連結角色，允許 Application Auto Scaling 呼叫 Neptune 和 CloudWatch，並代表您執行擴展。	2021 年 10 月 6 日
AWSApplicationAutoscalingRDSClusterPolicy – 新政策	新增 ElastiCache 的受管政策。此政策會連接到服務連結角色，允許 Application Auto Scaling 呼叫 ElastiCache 和 CloudWatch，並代表您執行擴展。	2021 年 8 月 19 日
Application Auto Scaling 開始追蹤變更	Application Auto Scaling 開始追蹤其 AWS 受管政策的變更。	2021 年 8 月 19 日

Application Auto Scaling 的服務連結角色

Application Auto Scaling 會將服務連結角色用於代表您呼叫其他 AWS 服務所需的許可。服務連結角色是直接連結至 AWS 服務的 AWS Identity and Access Management (IAM) 角色唯一類型。服務連結角色提供將許可委派給 AWS 服務的安全方式，因為只有連結的服務可以擔任服務連結角色。

對於與 Application Auto Scaling 整合的服務，Application Auto Scaling 會為您建立服務連結角色。每個服務都有一個服務連結角色。每個服務連結角色都信任由指定的服務委託人擔任其角色。如需詳細資訊，請參閱服務連結角色 ARN 參考。

Application Auto Scaling 包含每個服務連結角色的所有必要許可。這些受管許可由 Application Auto Scaling 建立和管理，並定義各種資源類型允許的動作。如需每個角色所授予許可的詳細資訊，請參閱 AWS Application Auto Scaling 的 受管政策。

目錄

建立服務連結角色所需的許可

Application Auto Scaling 需要許可，以便在您第一次 AWS 帳戶 `RegisterScalableTarget`呼叫指定服務的任何使用者時建立服務連結角色。如果目標服務沒有服務連結角色，Application Auto Scaling 會在您的帳戶中建立該角色。服務連結角色准許 Application Auto Scaling 代表您呼叫目標服務。

為了成功自動建立該角色，使用者必須有 `iam:CreateServiceLinkedRole` 動作的許可。

```
"Action": "iam:CreateServiceLinkedRole"
```

以下是授權為 Spot 機群建立服務連結角色的身分型政策。您可以在政策的 `Resource` 欄位中以 ARN 指定服務連結角色，以條件指定服務連結角色的服務委託人，如下所示。關於每個服務的 ARN，請參閱服務連結角色 ARN 參考。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "arn:aws:iam::*:role/aws-
service-role/ec2.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_EC2SpotFleetRequest",
            "Condition": {
                "StringLike": {
```

```
                    "iam:AWSServiceName":"ec2.application-
autoscaling.amazonaws.com"
                }
            }
        }
    ]
}
```

> **Note**
>
> iam:AWSServiceName IAM 條件金鑰指定角色所連接的服務委託人，此範例政策中以 *ec2.application-autoscaling*.amazonaws.com 表示。不要嘗試猜測服務委託人。若要檢視服務的服務委託人，請參閱AWS 服務 可與 Application Auto Scaling 搭配使用。

建立服務連結角色 (自動)

您不需要手動建立一個服務連結角色。當您呼叫 RegisterScalableTarget 時，Application Auto Scaling 會為您建立適當的服務連結角色。例如，若您為 Amazon ECS 服務設定自動擴展，則 Application Auto Scaling 會建立 AWSServiceRoleForApplicationAutoScaling_ECSService 角色。

建立服務連結角色 (手動)

若要建立服務連結角色，您可以使用 IAM 主控台 AWS CLI或 IAM API。如需詳細資訊，請參閱《IAM 使用者指南》中的建立服務連結角色。

建立服務連結角色 (AWS CLI)

使用下列 create-service-linked-role 命令來建立 Application Auto Scaling 服務連結角色。在請求中，指定服務名稱「字首」。

若要尋找服務名稱字首，相關資訊請參閱AWS 服務 可與 Application Auto Scaling 搭配使用一節，其中有每個服務的服務連結角色的服務委託人。服務名稱和服務委託人共用相同的字首。例如，若要建立 AWS Lambda 服務連結角色，請使用 lambda.application-autoscaling.amazonaws.com。

```
aws iam create-service-linked-role --aws-service-name prefix.application-
autoscaling.amazonaws.com
```

編輯服務連結角色

對於 Application Auto Scaling 建立的服務連結角色，您只能編輯其描述。如需詳細資訊，請參閱《IAM 使用者指南》中的編輯服務連結角色描述。

刪除服務連結角色

如果您不再對支援的服務使用 Application Auto Scaling，建議您刪除對應的服務連結角色。

您必須先刪除相關的 AWS 資源，才能刪除服務連結角色。這可避免您意外撤銷 Application Auto Scaling 使用資源的許可。如需詳細資訊，請參閱各項可擴展性資源的文件。例如，若要刪除 Amazon ECS 服務，請參閱《Amazon Elastic Container Service 開發人員指南》中的刪除 Amazon ECS 服務。

您可以使用 IAM 來刪除服務連結角色。如需詳細資訊，請參閱《IAM 使用者指南》中的刪除服務連結角色。

在您刪除服務連結角色後，Application Auto Scaling 會在您呼叫 `RegisterScalableTarget` 時再次建立角色。

Application Auto Scaling 服務連結角色的支援區域

Application Auto Scaling 支援在提供服務的所有 AWS 區域中使用服務連結角色。

服務連結角色 ARN 參考

下表列出適用於 Application Auto Scaling 的每個 服務連結角色的 AWS 服務 Amazon Resource Name (ARN)。

服務	ARN
AppStream 2.0	`arn:aws:iam:: 012345678910 :role/aws-service-role/appstream.application-autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_AppStreamFleet`
Aurora	`arn:aws:iam:: 012345678910 :role/aws-service-role/rds.application-autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_RDSCluster`

服務	ARN
Comprehend	arn:aws:iam:: *012345678910* :role/aws-service-role/comprehend.application-autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_ComprehendEndpoint
DynamoDB	arn:aws:iam:: *012345678910* :role/aws-service-role/dynamodb.application-autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_DynamoDBTable
ECS	arn:aws:iam:: *012345678910* :role/aws-service-role/ecs.application-autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_ECSService
ElastiCache	arn:aws:iam:: *012345678910* :role/aws-service-role/elasticache.application-autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_ElastiCacheRG
Keyspaces	arn:aws:iam:: *012345678910* :role/aws-service-role/cassandra.application-autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_CassandraTable
Lambda	arn:aws:iam:: *012345678910* :role/aws-service-role/lambda.application-autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_LambdaConcurrency
MSK	arn:aws:iam:: *012345678910* :role/aws-service-role/kafka.application-autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_KafkaCluster

服務	ARN
Neptune	arn:aws:iam:: *012345678910* :role/aws-service-role/neptune.application-autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_NeptuneCluster
SageMaker AI	arn:aws:iam:: *012345678910* :role/aws-service-role/sagemaker.application-autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_SageMakerEndpoint
Spot Fleets	arn:aws:iam:: *012345678910* :role/aws-service-role/ec2.application-autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_EC2SpotFleetRequest
WorkSpaces	arn:aws:iam:: *012345678910* :role/aws-service-role/workspaces.application-autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_WorkSpacesPool
自訂資源	arn:aws:iam:: *012345678910* :role/aws-service-role/custom-resource.application-autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_CustomResource

Note

您可以在 AWS CloudFormation 堆疊範本中為 AWS：：ApplicationAutoScaling：：ScalableTarget 資源的 RoleARN 屬性指定服務連結角色的 ARN，即使指定的服務連結角色尚不存在。 AWS::ApplicationAutoScaling::ScalableTarget Application Auto Scaling 會自動為您建立角色。

Application Auto Scaling 以身分為基礎的政策範例

根據預設，您 中的新使用者 AWS 帳戶 沒有執行任何動作的許可。IAM 管理員必須建立和指派 IAM 政策，它們可提供 IAM 身分 (例如使用者或角色)，以執行 Application Auto Scaling API 動作。

若要了解如何使用以下範例 JSON 政策文件來建立 IAM 政策，請參閱《IAM 使用者指南》中的在 JSON 標籤上建立政策。

目錄

Application Auto Scaling API 動作所需的許可

下列政策會在呼叫 Application Auto Scaling API 的常見使用案例下授予許可。制定身分型政策時，請參閱本節的相關資訊。每個政策會授予對所有或部分 Application Auto Scaling API 動作的許可。您還需要確定終端使用者具有使用目標服務和 CloudWatch 的許可 (如需詳細資訊，請參閱下一節)。

以下身分型政策會授予對所有 Application Auto Scaling API 動作的許可。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "application-autoscaling:*"
            ],
            "Resource": "*"
        }
    ]
}
```

以下身分型政策會授予在設定擴展政策時需要的所有 Application Auto Scaling API 動作的許可，而不是排定的動作。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "application-autoscaling:RegisterScalableTarget",
              "application-autoscaling:DescribeScalableTargets",
              "application-autoscaling:DeregisterScalableTarget",
              "application-autoscaling:PutScalingPolicy",
              "application-autoscaling:DescribeScalingPolicies",
              "application-autoscaling:DescribeScalingActivities",
              "application-autoscaling:DeleteScalingPolicy"
            ],
            "Resource": "*"
        }
    ]
}
```

以下身分型政策會授予在設定排定的動作時需要的所有 Application Auto Scaling API 動作的許可，而不是擴展政策。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "application-autoscaling:RegisterScalableTarget",
              "application-autoscaling:DescribeScalableTargets",
              "application-autoscaling:DeregisterScalableTarget",
              "application-autoscaling:PutScheduledAction",
              "application-autoscaling:DescribeScheduledActions",
              "application-autoscaling:DescribeScalingActivities",
              "application-autoscaling:DeleteScheduledAction"
            ],
```

```
            "Resource": "*"
        }
    ]
}
```

對目標服務和 CloudWatch 執行 API 動作所需的許可

為了成功設定 Application Auto Scaling 來用於目標服務，必須授予終端使用者許可來存取 Amazon CloudWatch 及他們將設定擴展的每個目標服務。使用以下政策授予存取目標服務和 CloudWatch 所需的最低許可。

目錄

AppStream 2.0 機群

以下身分型政策會授予對所有必要 AppStream 2.0 和 CloudWatch API 動作的許可。

JSON

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "appstream:DescribeFleets",
              "appstream:UpdateFleet",
              "cloudwatch:DescribeAlarms",
              "cloudwatch:PutMetricAlarm",
              "cloudwatch:DeleteAlarms"
            ],
            "Resource": "*"
        }
    ]
}
```

Aurora 複本

以下身分型政策會授予對所有必要 Aurora 和 CloudWatch API 動作的許可。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "rds:AddTagsToResource",
              "rds:CreateDBInstance",
              "rds:DeleteDBInstance",
              "rds:DescribeDBClusters",
              "rds:DescribeDBInstances",
              "cloudwatch:DescribeAlarms",
              "cloudwatch:PutMetricAlarm",
              "cloudwatch:DeleteAlarms"
            ],
            "Resource": "*"
        }
    ]
}
```

Amazon Comprehend 文件分類和實體識別器端點

以下身分型政策會授予對所有必要 Amazon Comprehend 和 CloudWatch API 動作的許可。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "comprehend:UpdateEndpoint",
              "comprehend:DescribeEndpoint",
              "cloudwatch:DescribeAlarms",
              "cloudwatch:PutMetricAlarm",
              "cloudwatch:DeleteAlarms"
            ],
            "Resource": "*"
        }
    ]
}
```

DynamoDB 資料表和全域次要索引

以下身分型政策會授予對所有必要 DynamoDB 和 CloudWatch API 動作的許可。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "dynamodb:DescribeTable",
              "dynamodb:UpdateTable",
              "cloudwatch:DescribeAlarms",
              "cloudwatch:PutMetricAlarm",
              "cloudwatch:DeleteAlarms"
            ],
```

```
            "Resource": "*"
        }
    ]
}
```

ECS 服務

以下身分型政策會授予對所有必要 ECS 和 CloudWatch API 動作的許可。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "ecs:DescribeServices",
              "ecs:UpdateService",
              "cloudwatch:DescribeAlarms",
              "cloudwatch:PutMetricAlarm",
              "cloudwatch:DeleteAlarms"
            ],
            "Resource": "*"
        }
    ]
}
```

ElastiCache 複寫群組

以下身分型政策會授予對所有必要 ElastiCache 和 CloudWatch API 動作的許可。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
```

```
                "elasticache:ModifyReplicationGroupShardConfiguration",
                "elasticache:IncreaseReplicaCount",
                "elasticache:DecreaseReplicaCount",
                "elasticache:DescribeReplicationGroups",
                "elasticache:DescribeCacheClusters",
                "elasticache:DescribeCacheParameters",
                "cloudwatch:DescribeAlarms",
                "cloudwatch:PutMetricAlarm",
                "cloudwatch:DeleteAlarms"
            ],
            "Resource": "*"
        }
    ]
}
```

Amazon EMR 叢集

以下身分型政策會授予對所有必要 Amazon EMR 和 CloudWatch API 動作的許可。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "elasticmapreduce:ModifyInstanceGroups",
                "elasticmapreduce:ListInstanceGroups",
                "cloudwatch:DescribeAlarms",
                "cloudwatch:PutMetricAlarm",
                "cloudwatch:DeleteAlarms"
            ],
            "Resource": "*"
        }
    ]
}
```

Amazon Keyspaces 資料表

以下身分型政策會授予對所有必要 Amazon Keyspaces 和 CloudWatch API 動作的許可。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "cassandra:Select",
              "cassandra:Alter",
              "cloudwatch:DescribeAlarms",
              "cloudwatch:PutMetricAlarm",
              "cloudwatch:DeleteAlarms"
            ],
            "Resource": "*"
        }
    ]
}
```

Lambda 函數

以下身分型政策會授予對所有必要 Lambda 和 CloudWatch API 動作的許可。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "lambda:PutProvisionedConcurrencyConfig",
              "lambda:GetProvisionedConcurrencyConfig",
              "lambda:DeleteProvisionedConcurrencyConfig",
              "cloudwatch:DescribeAlarms",
              "cloudwatch:PutMetricAlarm",
              "cloudwatch:DeleteAlarms"
            ],
            "Resource": "*"
        }
    ]
```

```
}
```

Amazon Managed Streaming for Apache Kafka (MSK) 代理程式儲存

以下身分型政策會授予對所有必要 Amazon MSK 和 CloudWatch API 動作的許可。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "kafka:DescribeCluster",
              "kafka:DescribeClusterOperation",
              "kafka:UpdateBrokerStorage",
              "cloudwatch:DescribeAlarms",
              "cloudwatch:PutMetricAlarm",
              "cloudwatch:DeleteAlarms"
            ],
            "Resource": "*"
        }
    ]
}
```

Neptune 叢集

以下身分型政策會授予對所有必要 Neptune 和 CloudWatch API 動作的許可。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "rds:AddTagsToResource",
              "rds:CreateDBInstance",
```

```
                "rds:DescribeDBInstances",
                "rds:DescribeDBClusters",
                "rds:DescribeDBClusterParameters",
                "rds:DeleteDBInstance",
                "cloudwatch:DescribeAlarms",
                "cloudwatch:PutMetricAlarm",
                "cloudwatch:DeleteAlarms"
            ],
            "Resource": "*"
        }
    ]
}
```

SageMaker AI 端點

下列身分型政策會將許可授予所有必要的 SageMaker AI 和 CloudWatch API 動作。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "sagemaker:DescribeEndpoint",
                "sagemaker:DescribeEndpointConfig",
                "sagemaker:DescribeInferenceComponent",
                "sagemaker:UpdateEndpointWeightsAndCapacities",
                "sagemaker:UpdateInferenceComponentRuntimeConfig",
                "cloudwatch:DescribeAlarms",
                "cloudwatch:PutMetricAlarm",
                "cloudwatch:DeleteAlarms"
            ],
            "Resource": "*"
        }
    ]
}
```

Spot 機群 (Amazon EC2)

以下身分型政策會授予對所有必要 Spot 機群 和 CloudWatch API 動作的許可。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "ec2:DescribeSpotFleetRequests",
              "ec2:ModifySpotFleetRequest",
              "cloudwatch:DescribeAlarms",
              "cloudwatch:PutMetricAlarm",
              "cloudwatch:DeleteAlarms"
            ],
            "Resource": "*"
        }
    ]
}
```

自訂資源

以下身分型政策會授予 API Gateway API 執行動作的許可。此政策還會授予對所有必要 CloudWatch 動作的許可。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "execute-api:Invoke",
              "cloudwatch:DescribeAlarms",
              "cloudwatch:PutMetricAlarm",
              "cloudwatch:DeleteAlarms"
```

```
            ],
            "Resource": "*"
        }
    ]
}
```

在 中工作的許可 AWS Management Console

沒有獨立的 Application Auto Scaling 主控台。與 Application Auto Scaling 整合的大部分服務都有一些功能，專門協助您透過主控台來設定擴展。

在大多數情況下，每個服務都會提供 AWS 受管 （預先定義） IAM 政策，其定義對 主控台的存取，其中包含 Application Auto Scaling API 動作的許可。如需詳細資訊，請參閱您要使用其主控台之服務的文件。

您也可以建立自己的自訂 IAM 政策，以給予使用者精細許可在 AWS Management Console檢視和使用特定的 Application Auto Scaling API 動作。您可以使用先前章節中的範例政策；不過，這些政策是專為使用 AWS CLI 或 SDK 提出的請求而設計。主控台會針對其功能使用其他的 API 動作，所以這些政策可能不會如預期般運作。例如，若要設定步驟擴展，使用者可能需要額外許可來建立和管理 CloudWatch 警示。

> **Tip**
>
> 在主控台中執行工作時，為協助找出所需的 API 動作，您可以使用像是 AWS CloudTrail的服務。如需詳細資訊，請參閱「AWS CloudTrail 使用者指南」。

以下身分型政策會授予為 Spot 機群設定擴展政策的許可。除了 Spot 機群的 IAM 許可之外，從 Amazon EC2 主控台存取機群擴展設定的主控台使用者還必須取得適當的許可，以存取支援動態擴展的服務。

JSON

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
```

```
            "Action": [
                "application-autoscaling:*",
                "ec2:DescribeSpotFleetRequests",
                "ec2:ModifySpotFleetRequest",
                "cloudwatch:DeleteAlarms",
                "cloudwatch:DescribeAlarmHistory",
                "cloudwatch:DescribeAlarms",
                "cloudwatch:DescribeAlarmsForMetric",
                "cloudwatch:GetMetricStatistics",
                "cloudwatch:ListMetrics",
                "cloudwatch:PutMetricAlarm",
                "cloudwatch:DisableAlarmActions",
                "cloudwatch:EnableAlarmActions",
                "sns:CreateTopic",
                "sns:Subscribe",
                "sns:Get*",
                "sns:List*"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "arn:aws:iam::*:role/aws-
service-role/ec2.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_EC2SpotFleetRequest",
            "Condition": {
                "StringLike": {
                    "iam:AWSServiceName":"ec2.application-
autoscaling.amazonaws.com"
                }
            }
        }
    ]
}
```

此政策允許主控台使用者在 Amazon EC2 主控台檢視和修改擴展政策，以及在 CloudWatch 主控台建立和管理 CloudWatch 警示。

您可以調整 API 動作以限制使用者存取。例如，將 `application-autoscaling:*` 替換為 `application-autoscaling:Describe*` 表示使用者具有唯讀存取權。

您也可以視需要調整 CloudWatch 許可，以限制使用者對 CloudWatch 功能的存取。如需詳細資訊，請參閱《Amazon CloudWatch 使用者指南》中的 CloudWatch 主控台所需的許可。 Amazon CloudWatch

Application Auto Scaling 存取的疑難排解

如果您在使用 Application Auto Scaling 時遇到 `AccessDeniedException` 或類似困難，請參閱本節的資訊。

我未獲授權在 Application Auto Scaling 中執行動作

如果您在呼叫 AWS API 操作`AccessDeniedException`時收到 ，表示您正在使用的 AWS Identity and Access Management (IAM) 登入資料沒有發出該呼叫所需的許可。

當 `mateojackson` 使用者嘗試檢視可擴展目標的詳細資訊，但沒有 `application-autoscaling:DescribeScalableTargets` 許可時，便會發生以下範例錯誤。

```
An error occurred (AccessDeniedException) when calling the DescribeScalableTargets
 operation: User: arn:aws:iam::123456789012:user/mateojackson is not authorized to
 perform: application-autoscaling:DescribeScalableTargets
```

如果您遇到此錯誤或類似錯誤，則必須聯絡管理員以取得協助。

您帳戶的管理員必須確定您具有許可，可存取 Application Auto Scaling 在目標服務和 CloudWatch 中存取資源時使用的所有 API 動作。所需的許可視您使用的資源而有所不同。當使用者首次為給定的資源設定擴展時，Application Auto Scaling 也需要有許可來建立服務連結角色。

我是管理員，我的 IAM 政策傳回錯誤或無法如預期般運作

除了 Application Auto Scaling 動作外，您的 IAM 政策還必須授予許可來呼叫目標服務和 CloudWatch。如果使用者或應用程式沒有這些額外許可，其存取作業可能會意外遭到拒絕。若要為您帳戶中的使用者和應用程式撰寫 IAM 政策，請參閱 Application Auto Scaling 以身分為基礎的政策範例中的資訊。

如需如何執行驗證的相關資訊，請參閱目標資源上 Application Auto Scaling API 呼叫的許可驗證。

請注意，某些許可問題也可能起因於建立 Application Auto Scaling 所使用的服務連結角色時發生問題。如需有關建立這些服務連結角色的詳細資訊，請參閱 Application Auto Scaling 的服務連結角色。

目標資源上 Application Auto Scaling API 呼叫的許可驗證

向 Application Auto Scaling API 動作提出授權請求需要 API 發起人具有存取目標服務和 CloudWatch 中 AWS 資源的許可。在繼續處理與目標服務和 CloudWatch 相關聯的請求之前，Application Auto Scaling 會先驗證請求的許可。為了這樣做，我們發出一系列呼叫來驗證目標資源上的 IAM 許可。傳回的回應由 Application Auto Scaling 讀取。如果 IAM 許可不允許指定的動作，則 Application Auto Scaling 的請求會失敗，並傳回錯誤給使用者，其中包含缺少許可的相關資訊。這可確保使用者想要部署的擴展組態如預期般運作，並於請求失敗時傳回有用的錯誤。

為了示範運作過程，以下資訊提供有關 Application Auto Scaling 如何對 Aurora 和 CloudWatch 執行許可驗證的詳細資訊。

當使用者對 Aurora 資料庫叢集呼叫 `RegisterScalableTarget` API 時，Application Auto Scaling 會執行下列所有檢查，以確認使用者具有必要的許可 (以粗體顯示)。

- rds:CreateDBInstance：為了判斷使用者是否具有此許可，我們對 `CreateDBInstance` API 操作傳送請求，嘗試在使用者指定的 Aurora 資料庫叢集中以無效參數 (空的執行個體 ID) 建立資料庫執行個體。對於獲授權的使用者，API 在稽核請求後會傳回 `InvalidParameterValue` 錯誤碼回應。但是，對於未經授權的使用者，我們會遇到 `AccessDenied` 錯誤，且 Application Auto Scaling 請求會失敗，並將 `ValidationException` 錯誤傳給使用者，其中列出缺少的許可。
- rds:DeleteDBInstance：我們將空的執行個體 ID 傳送給 `DeleteDBInstance` API 操作。對於獲授權的使用者，此請求會導致 `InvalidParameterValue` 錯誤。對於未經授權的使用者，則會導致 `AccessDenied` 並傳送驗證例外給使用者（與第一個項目符號所述的處理相同）。
- rds:AddTagsToResource：因為 `AddTagsToResource` API 操作需要 Amazon Resource Name (ARN)，所以必須使用無效帳戶 ID (12345) 指定「虛擬」資源，並指定虛擬執行個體 ID (non-existing-db)，以建構 ARN (`arn:aws:rds:us-east-1:12345:db:non-existing-db`)。對於獲授權的使用者，此請求會導致 `InvalidParameterValue` 錯誤。對於未經授權的使用者，則會導致 `AccessDenied` 並傳送驗證例外給使用者。
- rds：DescribeDBClusters：我們會描述要註冊自動擴展之資源的叢集名稱。對於獲授權的使用者，我們得到有效的描述結果。對於未經授權的使用者，則會導致 `AccessDenied` 並傳送驗證例外給使用者。
- rds：DescribeDBInstances：我們使用篩選條件來呼叫 `DescribeDBInstances` API，該`db-cluster-id`篩選條件會篩選使用者提供的叢集名稱，以註冊可擴展的目標。對於獲授權的使用者，我們獲准描述資料庫叢集中的所有資料庫執行個體。對於未經授權的使用者，此呼叫會導致 `AccessDenied` 並傳送驗證例外給使用者。

- cloudwatch:PutMetricAlarm：我們呼叫 `PutMetricAlarm` API 且不帶任何參數。因為缺少警示名稱，對於獲授權的使用者，此請求會導致 `ValidationError`。對於未經授權的使用者，則會導致 `AccessDenied` 並傳送驗證例外給使用者。
- cloudwatch:DescribeAlarms：我們呼叫 `DescribeAlarms` API，並將記錄數上限值設為 1。對於獲授權的使用者，我們預期回應中有一個警示的資訊。對於未經授權的使用者，此呼叫會導致 `AccessDenied` 並傳送驗證例外給使用者。
- cloudwatch:DeleteAlarms：類似於上方的 `PutMetricAlarm`，我們不提供參數給 `DeleteAlarms` 請求。因為請求中缺少警示名稱，對於獲授權的使用者，此呼叫會失敗並傳回 `ValidationError`。對於未經授權的使用者，則會導致 `AccessDenied` 並傳送驗證例外給使用者。

發生上述任何一個驗證例外都會記錄下來。您可以採取步驟，使用 手動識別哪些呼叫驗證失敗 AWS CloudTrail。如需詳細資訊，請參閱「AWS CloudTrail 使用者指南」。

> **Note**
>
> 如果您使用 CloudTrail 收到有關 Application Auto Scaling 事件的提醒，則這些提醒會包含預設情況下用於驗證使用者許可的 Application Auto Scaling 呼叫。要篩選出這些提醒，請使用 `invokedBy` 欄位，該欄位會包含進行這些驗證檢查的 `application-autoscaling.amazonaws.com`。

使用介面 VPC 端點存取 Application Auto Scaling

您可以使用 在 VPC 和 Application Auto Scaling 之間 AWS PrivateLink 建立私有連線。您可以像在 VPC 中一樣存取 Application Auto Scaling，無需使用網際網路閘道、NAT 裝置、VPN 連接或 AWS Direct Connect 連線。VPC 中的執行個體不需要公有 IP 地址即可存取 Application Auto Scaling。

您可以建立由 AWS PrivateLink提供支援的介面端點來建立此私有連線。我們會在您為介面端點啟用的每個子網中建立端點網路介面。這些是申請者管理的網路介面，可做為目的地為 Application Auto Scaling 之流量的進入點。

如需詳細資訊，請參閱《 AWS PrivateLink 指南》中的AWS 服務 透過 存取 AWS PrivateLink 。

目錄

建立介面 VPC 端點

使用下列服務名稱為 Application Auto Scaling 建立端點：

```
com.amazonaws.region.application-autoscaling
```

如需詳細資訊，請參閱《 AWS PrivateLink 指南》中的使用介面 VPC 端點存取 AWS 服務。

您不需要變更任何其他設定。Application Auto Scaling 會使用 AWS 服務端點或私有介面 VPC 端點呼叫其他服務，以使用中者為準。

建立 VPC 端點政策

您可以將政策連接到 VPC 端點，以控制對 Application Auto Scaling API 的存取。此政策指定：

- 可執行動作的委託人。
- 可執行的動作。
- 可供執行動作的資源。

下列範例顯示 VPC 端點政策，拒絕所有人透過端點刪除擴展政策的許可。範例政策也會授予所有人執行所有其他動作的許可。

```
{
   "Statement": [
        {
            "Action": "*",
            "Effect": "Allow",
            "Resource": "*",
            "Principal": "*"
        },
        {
            "Action": "application-autoscaling:DeleteScalingPolicy",
            "Effect": "Deny",
            "Resource": "*",
            "Principal": "*"
        }
    ]
}
```

如需詳細資訊，請參閱《AWS PrivateLink 指南》中的 VPC 端點政策。

Application Auto Scaling 的恢復能力

AWS 全球基礎設施是以 AWS 區域和可用區域為基礎建置。

AWS 區域提供多個實體分隔和隔離的可用區域，這些可用區域以低延遲、高輸送量和高備援聯網連接。

透過可用區域，您可以設計與操作的應用程式和資料庫，在可用區域之間自動容錯移轉而不會發生中斷。可用區域的可用性、容錯能力和擴展能力，均較單一或多個資料中心的傳統基礎設施還高。

如需 AWS 區域和可用區域的詳細資訊，請參閱 AWS 全球基礎設施。

Application Auto Scaling 中的基礎設施安全

Application Auto Scaling 為受管服務，受到 AWS 全球網路安全的保護。如需 AWS 安全服務以及如何 AWS 保護基礎設施的資訊，請參閱AWS 雲端安全。若要使用基礎設施安全的最佳實務設計您的 AWS 環境，請參閱安全支柱 AWS Well-Architected Framework 中的基礎設施保護。

您可以使用 AWS 發佈的 API 呼叫，透過網路存取 Application Auto Scaling。使用者端必須支援下列專案：

- Transport Layer Security (TLS)。我們需要 TLS 1.2 並建議使用 TLS 1.3。
- 具備完美轉送私密(PFS)的密碼套件，例如 DHE (Ephemeral Diffie-Hellman)或 ECDHE (Elliptic Curve Ephemeral Diffie-Hellman)。現代系統(如 Java 7 和更新版本)大多會支援這些模式。

此外，請求必須使用存取金鑰 ID 和與 IAM 主體相關聯的私密存取金鑰來簽署。或者，您可以透過 AWS Security Token Service (AWS STS) 來產生暫時安全憑證來簽署請求。

Application Auto Scaling 的合規驗證

若要了解 是否 AWS 服務 在特定合規計劃的範圍內，請參閱AWS 服務 合規計劃範圍內然後選擇您感興趣的合規計劃。如需一般資訊，請參閱 AWS Compliance Programs。

您可以使用 下載第三方稽核報告 AWS Artifact。如需詳細資訊，請參閱在 中下載報告 AWS Artifact。

您使用 時的合規責任 AWS 服務 取決於資料的機密性、您公司的合規目標，以及適用的法律和法規。AWS 提供下列資源來協助合規：

- 安全合規與治理 - 這些解決方案實作指南內容討論了架構考量，並提供部署安全與合規功能的步驟。

- HIPAA 合格服務參考 – 列出 HIPAA 合格服務。並非所有 AWS 服務 都符合 HIPAA 資格。
- AWS 合規資源 – 此工作手冊和指南集合可能適用於您的產業和位置。
- AWS 客戶合規指南 – 透過合規的角度了解共同責任模型。本指南摘要說明跨多個架構 （包括國家標準技術研究所 (NIST)、支付卡產業安全標準委員會 (PCI) 和國際標準化組織 (ISO)) 保護 AWS 服務和映射指南至安全控制的最佳實務。
- 《AWS Config 開發人員指南》中的使用規則評估資源 – AWS Config 服務會評估資源組態符合內部實務、產業準則和法規的程度。
- AWS Security Hub – 這 AWS 服務 可讓您全面檢視其中的安全狀態 AWS。Security Hub 使用安全控制，可評估您的 AWS 資源並檢查您的法規遵循是否符合安全業界標準和最佳實務。如需支援的服務和控制清單，請參閱「Security Hub 控制參考」。
- Amazon GuardDuty – 這會監控您的環境是否有可疑和惡意活動，以 AWS 服務 偵測對您 AWS 帳戶、工作負載、容器和資料的潛在威脅。GuardDuty 可滿足特定合規架構所規定的入侵偵測需求，以協助您因應 PCI DSS 等各種不同的合規需求。
- AWS Audit Manager – 這 AWS 服務 可協助您持續稽核 AWS 用量，以簡化您管理風險和符合法規和業界標準的方式。

Application Auto Scaling 的配額

您的 AWS 帳戶 具有預設配額，先前稱為每個配額的限制 AWS 服務。除非另有說明，否則每個配額都是區域特定規定。您可以請求提高某些配額，而其他配額無法提高。

若要檢視 AdApplication Auto Scaling 的配額，請開啟 Service Quotas 主控台。在導覽窗格中，請選擇 AWS 服務，然後選取 Application Auto Scaling。

若要請求提升配額，請參閱《Service Quotas 使用者指南》中的請求提升配額。

您的 AWS 帳戶 具有下列與 Application Auto Scaling 相關的配額。

名稱	預設	可調整
每個資源類型的可擴展目標	Amazon DynamoDB：5，000 | Amazon ECS：3，000 | Amazon Keyspaces：1，500 | 其他資源類型：500	是
每個可擴展目標的擴展政策（步驟擴展和目標追蹤政策）	50	否
每個可擴展目標的排程動作	200	否
每個步驟擴展政策的步驟調整數量	20	是

在擴增工作負載時，請記住服務配額。例如，當您達到服務允許的容量單位數目上限時，向外擴展將會停止。如果需求下降且目前容量減少，則 Application Auto Scaling 可以再次水平擴展。若要避免再次達到此容量限額，您可以請求提升額度。每個服務都有自己的預設配額，以供資源的最大容量使用。關於其他 Amazon Web Services 的預設配額，相關資訊請參閱 Amazon Web Services 一般參考 中的服務端點和配額。

Application Auto Scaling 的文件歷史記錄

下表說明 Application Auto Scaling 文件自 2018 年 1 月起的重要增補。如需有關此文件更新的通知，您可以訂閱 RSS 摘要。

變更	描述	日期
新增對 ElastiCache Memcached 叢集的支援	使用 Application Auto Scaling 水平擴展 Memcached 叢集的節點數量。如需詳細資訊，請參閱 ElastiCache 和 Application Auto Scaling。	2025 年 4 月 10 日
AWS 受管政策更新	Application Auto Scaling 已更新`AWSApplicationAutoscalingElastiCacheRGPolicy` 政策。	2025 年 4 月 10 日
指南變更	Application Auto Scaling 使用者指南中的新主題可協助您開始使用預測擴展搭配 Application Auto Scaling。請參閱 Application Auto Scaling 預測擴展。	2024 年 11 月 21 日
AWS 受管政策更新	Application Auto Scaling 已更新`AWSApplicationAutoscalingECSServicePolicy` 政策。	2024 年 11 月 21 日
新增對 WorkSpaces 集區的支援	使用 Application Auto Scaling 擴展 WorkSpaces 集區。如需詳細資訊，請參閱 Amazon WorkSpaces 和 Application Auto Scaling。Application Auto Scaling 對 AWS 受管政策的更新主題已更新，以列出與	2024 年 6 月 27 日

	WorkSpaces 整合的新受管政策。	
指南變更	已更新配額文件中的每種資源類型可擴展目標的數量上限項目。請參閱 Application Auto Scaling 的配額。	2024 年 1 月 16 日
支援 SageMaker AI 推論元件	使用 Application Auto Scaling 擴展推論元件的副本。	2023 年 11 月 29 日
AWS 受管政策更新	Application Auto Scaling 已更新`AWSApplicationAutoscalingSageMakerEndpointPolicy` 政策。	2023 年 11 月 13 日
支援 SageMaker AI Serverless 佈建並行	使用 Application Auto Scaling 擴展無伺服器端點的佈建並行。	2023 年 5 月 9 日
使用標籤對可擴展目標進行分類	您現在可以用標籤的形式將中繼資料指派給 Application Auto Scaling 可擴展目標。請參閱 Application Auto Scaling 的標籤支援。	2023 年 3 月 20 日
CloudWatch 指標數學的支援	建立目標追蹤擴展政策時，您現在可以使用指標數學。利用指標數學，可查詢多個 CloudWatch 指標，並使用數學表達式根據這些指標來建立新的時間序列。請參閱使用指標數學運算建立 Application Auto Scaling 的目標追蹤擴展政策	2023 年 3 月 14 日

不擴展的原因	您現在可以擷取 Application Auto Scaling 未使用 Application Auto Scaling API 擴展您資源的可機讀原因。請參閱 Application Auto Scaling 擴展活動。	2023 年 1 月 4 日
指南變更	已更新配額文件中的每種資源類型可擴展目標的數量上限項目。請參閱 Application Auto Scaling 的配額。	2022 年 5 月 6 日
新增對 Amazon Neptune 叢集的支援	使用 Application Auto Scaling 來擴展 Amazon Neptune 資料庫叢集中的複本數量。如需詳細資訊，請參閱 Amazon Neptune 和 Application Auto Scaling。Application Auto Scaling 新增 AWS 受管政策的主題內容更新，列出了與 Neptune 整合的新受管政策。	2021 年 10 月 6 日
Application Auto Scaling 現在會報告其 AWS 受管政策的變更	從 2021 年 8 月 19 日開始，受管政策的變更會在 Application Auto Scaling 更新 AWS 受管政策主題中報告。列出的第一個變更是新增 ElastiCache (Redis OSS) 所需的許可。	2021 年 8 月 19 日
新增對 ElastiCache (Redis OSS) 複寫群組的支援	使用 Application Auto Scaling 來擴展 ElastiCache (Redis OSS) 複寫群組（叢集）的節點群組數量和每個節點群組的複本數量。如需詳細資訊，請參閱 ElastiCache (Redis OSS) 和 Application Auto Scaling。	2021 年 8 月 19 日

指南變更	Application Auto Scaling 使用者指南中有新的 IAM 主題，可協助您對 Application Auto Scaling 的存取進行疑難排解。如需詳細資訊，請參閱適用於 Application Auto Scaling 的 Identity and Access Management。對於目標服務和 Amazon CloudWatch 上的動作，也新增 IAM 許可政策範例。如需詳細資訊，請參閱使用 AWS CLI 或 SDK 的範例政策。	2021 年 2 月 23 日
新增支援當地時區	您現在可以在當地時區建立排定的動作。如果您的時區遵守日光節約時間，則會依據日光節約時間 (DST) 自動調整。如需詳細資訊，請參閱排程擴展。	2021 年 2 月 2 日
指南變更	《Application Auto Scaling 使用者指南》中有新的教學課程，可協助您了解如何使用目標追蹤擴展政策和排程擴展，以便使用 Application Auto Scaling 時提高應用程式的可用性。	2020 年 10 月 15 日
新增支援 Amazon Managed Streaming for Apache Kafka 叢集儲存	使用目標追蹤擴展政策來水平擴展與 Amazon MSK 叢集相關聯的代理程式儲存數量。	2020 年 9 月 30 日
新增支援 Amazon Comprehend 實體辨識器端點	使用 Application Auto Scaling 來擴展佈建給 Amazon Comprehend 實體辨識器端點的推論單位數。	2020 年 9 月 28 日

新增 Amazon Keyspaces (適用於 Apache Cassandra) 資料表的支援	使用 Application Auto Scaling 擴展 Amazon Keyspaces 資料表的佈建輸送量 (讀和寫容量)。	2020 年 4 月 23 日
新增「安全」章節	《Application Auto Scaling 使用者指南》中有新的安全性章節，可協助您了解如何在使用 Application Auto Scaling 時套用共同責任模型。在此更新中，使用者指南的〈驗證和存取控制〉一章換成更有用的新小節：適用於 Application Auto Scaling 的 Identity and Access Management。	2020 年 1 月 16 日
次要更新	各種改進和修正。	2020 年 1 月 15 日
新增通知功能	Application Auto Scaling 現在會傳送事件至 Amazon EventBridge，並在發生特定動作 AWS Health Dashboard 時通知您。如需詳細資訊，請參閱 Application Auto Scaling 監控。	2019 年 12 月 20 日
新增對 AWS Lambda 函數的支援	使用 Application Auto Scaling 擴展 Lambda 函數的佈建並行。	2019 年 12 月 3 日
新增對 Amazon Comprehend 文件分類端點的支援	使用 Application Auto Scaling 擴展 Amazon Comprehend 文件分類端點的輸送容量。	2019 年 11 月 25 日
目標追蹤擴展政策增加支援 AppStream 2.0	使用目標追蹤擴展政策來擴展 AppStream 2.0 機群的大小。	2019 年 11 月 25 日

支援 Amazon VPC 端點	您現在可以在 VPC 和 Application Auto Scaling 之間建立私有連線。如需遷移考量和指示，請參閱 Application Auto Scaling 和界面 VPC 端點。	2019 年 11 月 22 日
暫停和恢復擴展	新增對暫停和繼續擴展的支援。如需詳細資訊，請參閱暫停和繼續擴展 Application Auto Scaling。	2019 年 8 月 29 日
指南變更	改進 Application Auto Scaling 文件中的排程擴展、步驟擴展政策和目標追蹤擴展政策這三節。	2019 年 3 月 11 日
新增對自訂資源的支援	使用 Application Auto Scaling 擴展由您自己的應用程式或服務所提供的自訂資源。更多詳細資訊，請參閱我們的 GitHub 儲存庫。	2018 年 7 月 9 日
新增對 SageMaker AI 端點變體的支援	使用 Application Auto Scaling 來擴展佈建給變體的端點執行個體數。	2018 年 2 月 28 日

下表說明 Application Auto Scaling 文件在 2018 年 1 月前的重要變更。

變更	描述	日期
新增對 Aurora 複本的支援	使用 Application Auto Scaling 擴展所需的計數。如需詳細資訊，請參閱《Amazon RDS 使用者指南》中的使用 Amazon Aurora Auto Scaling 搭配 Aurora 複本。	2017 年 11 月 17 日

變更	描述	日期
新增對排程擴展功能的支援	使用排程擴展功能，在特定的預設時間或間隔進行資源的擴展。如需詳細資訊，請參閱 Application Auto Scaling 的排程擴展。	2017 年 11 月 8 日
新增對目標追蹤擴展政策的支援	使用目標追蹤擴展政策，只需幾個簡單步驟即可設定應用程式動態擴展。如需詳細資訊，請參閱 Application Auto Scaling 的目標追蹤擴展政策。	2017 年 7 月 12 日
對 DynamoDB 資料表及全域次要索引新增支援佈建的讀和寫容量	使用 Application Auto Scaling 來擴展佈建輸送量 (讀和寫容量)。如需詳細資訊，請參閱《Amazon DynamoDB 開發人員指南》中的使用 DynamoDB Auto Scaling 管理輸送容量。	2017 年 6 月 14 日
新增支援 AppStream 2.0 機群	使用 Application Auto Scaling 擴展機群的大小。如需詳細資訊，請參閱《Amazon AppStream 2.0 管理指南》中的 AppStream 2.0 的機群自動擴展。	2017 年 3 月 23 日
新增支援 Amazon EMR 叢集	使用 Application Auto Scaling 擴展核心和任務節點。如需詳細資訊，請參閱《Amazon EMR 管理指南》中的在 Amazon EMR 中使用自動擴展。	2016 年 11 月 18 日

變更	描述	日期
新增對 Spot 機群的支援	使用 Application Auto Scaling 擴展目標容量。如需詳細資訊，請參閱《Amazon EC2 使用者指南》中的 Spot 機群的自動擴展。	2016 年 9 月 1 日
新增支援 Amazon ECS 服務	使用 Application Auto Scaling 擴展所需的計數。如需詳細資訊，請參閱《Amazon Elastic Container Service 開發人員指南》中的服務自動擴展。	2016 年 8 月 9 日

本文為英文版的機器翻譯版本，如內容有任何歧義或不一致之處，概以英文版為準。